POLITEKNIK SANDAKAN SABAH

ARDUINO PROJECT FOR BEGINNERS

for AProB-Kit User

Basic Robotic and Automation System

- 10 Basic Arduino Projects
- QR Code for Project Video
- Circuit Diagram
- Arduino IDE Code

NORINA BINTI YADIN
NAJWA SHAHIDA BINTI MOHAMAD
NOOR AZLYN BINTI AB GHAFAR

pss.polycc.edu.com

POLITEKNIK SANDAKAN SABAH

ARDUINO PROJECT FOR BEGINNERS

NORINA BINTI YADIN
NAJWA SHAHIDA BINTI MOHAMAD
NOOR AZLYN BINTI AB GHAFAR

pss.polycc.edu.com

Published by:
Politeknik Sandakan Sabah
Education Hub, Batu 10,
Jalan Sungai Batang,
90000 Sandakan Sabah

pss.mypolycc.edu.my

First Published 2023

eISBN:

e ISBN 978-629-98067-6-9

9 786299 806769

A catalogue record for this e-book is available from the Politeknik Sandakan Sabah Website.
http://surl.li/nsscd

PREFACE

Arduino Project for Beginners First Edition is specially written for Politeknik Sandakan Sabah students who are undergoing Diploma in Agrotechnology and Diploma in Technology Aquaculture.

It contains ten (10) Arduino projects with their respective uses. Each Arduino project contains a list of components required, QR code for video, circuit diagram, and Arduino IDE code.

In this book, fourteen (14) components are explained clearly. Pictures of each component are included for easy identification and understanding.

Arduino Uno board was chosen as microcontroller because it is easy to use. It uses a simplified version of C/C++ language which is easily adaptable. Arduino Uno boards are available at a low cost and used Arduino IDE to write the program which is freely available.

We hope this book will help students to master basic projects in the use of Arduino Uno as microcontroller and develop the knowledge gained to a higher level.

Norina Binti Yadin
Najwa Shahida Binti Mohamad
Noor Azlyn Binti AB Ghafar

TABLE OF CONTENTS

TABLE OF CONTENTS

TOPIC

A

ARDUINO UNO

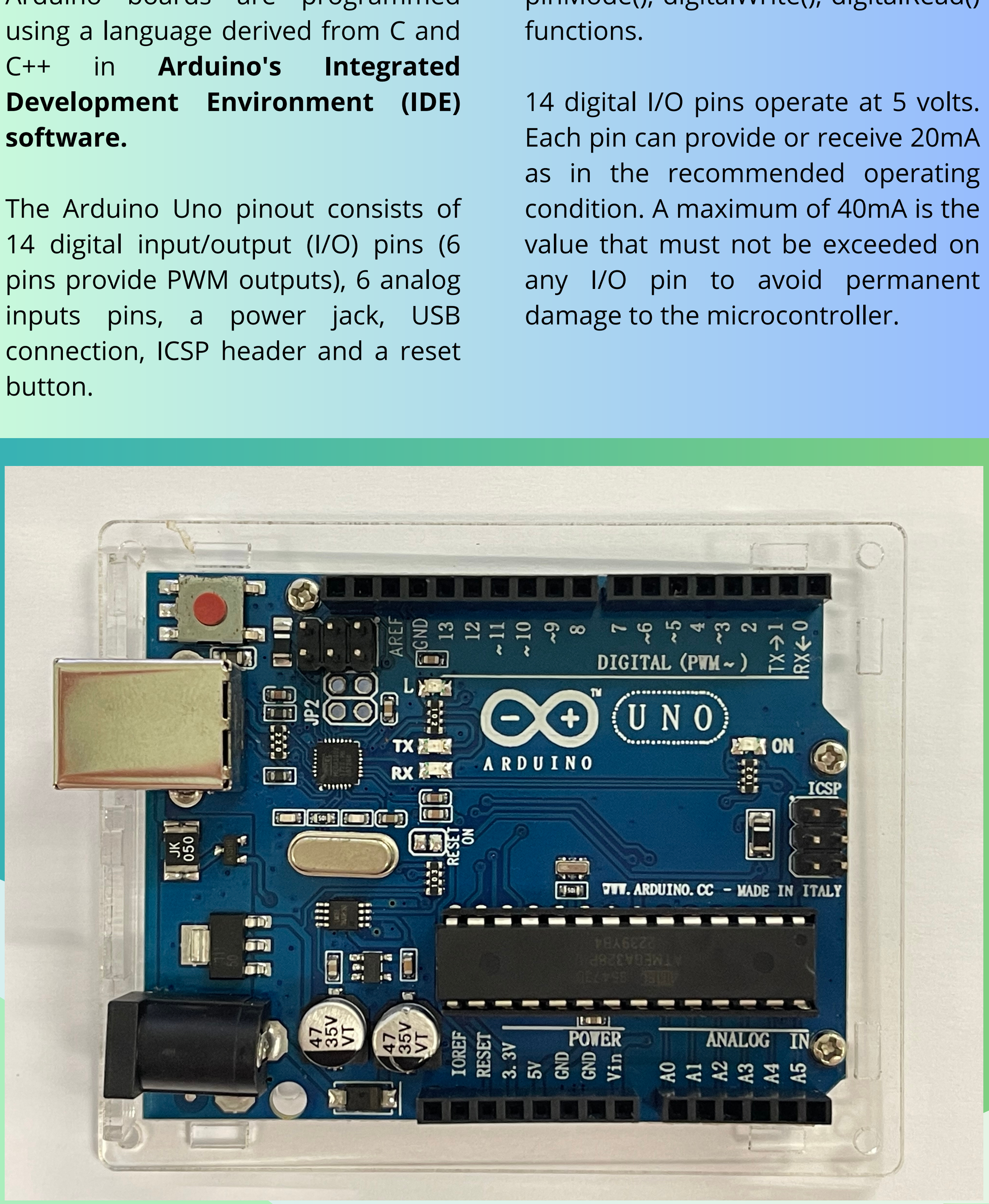
Arduino Uno is microcontroller board based on the ATmega328P.

Arduino boards are programmed using a language derived from C and C++ in **Arduino's Integrated Development Environment (IDE) software.**

The Arduino Uno pinout consists of 14 digital input/output (I/O) pins (6 pins provide PWM outputs), 6 analog inputs pins, a power jack, USB connection, ICSP header and a reset button.

Each of the 14 digital I/O pins labeled 0 through 13 on the board can be used as an input or output, using pinMode(), digitalWrite(), digitalRead() functions.

14 digital I/O pins operate at 5 volts. Each pin can provide or receive 20mA as in the recommended operating condition. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.



Arduino Uno Board

14 digital I/O pins have specialized functions as follow:

- **Pin 13 (LED)** - There is a build-in LED driven by digital pin 13. When the pin is HIGH value, the built-in LED is on, when the pin is LOW value, it's off.

- **PWM Pins** - Pins 3, 5, 6, 9, 10 and 11 are PWM pins. It provides an 8-bit PWM output with the analogWrite() function.

- **Serial: 0 (RX) and 1 (TX)** - Used to receive (RX) and transmit (TX) TTL serial data. Serial communication is used to exchange data between the Arduino board and another serial such as computers, displays, sensors and more.

Digital I/O pins are a way of representing voltage in 1 bit: either 0 or 1. Digital pins are either on or off. When on they are in a high voltage state of 5V and when off they are in a low voltage state of 0V.

There are 6 **Pulse Width Modulation (PWM) pins**. PWM is a modulation technique used to encode a message into a pulsing signal.

A PWM comprises two key components, which are frequency and duty cycle.

The PWM frequency dictates how long it takes to complete a single cycle (period) and how quickly the signal fluctuates from high to low.

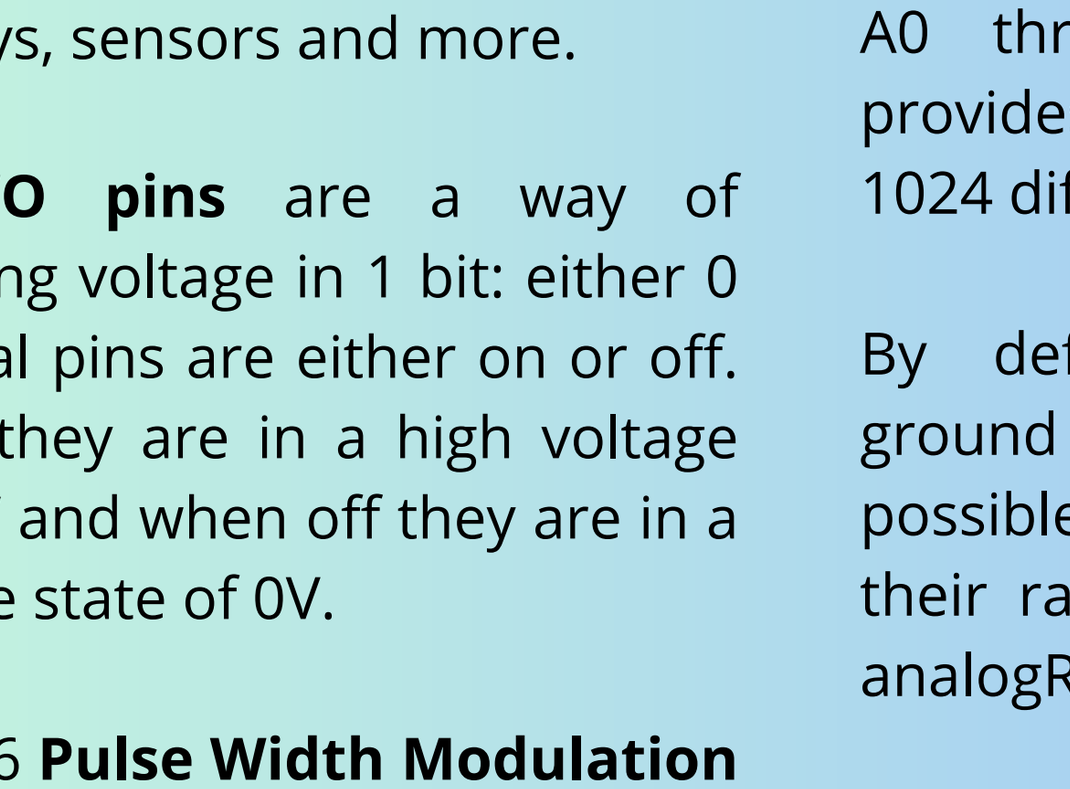
The duty cycle determines how long a signal stays high out of the total period. Duty cycle is represented in percentage. Duty cycle changes according to the parameters set by the user.

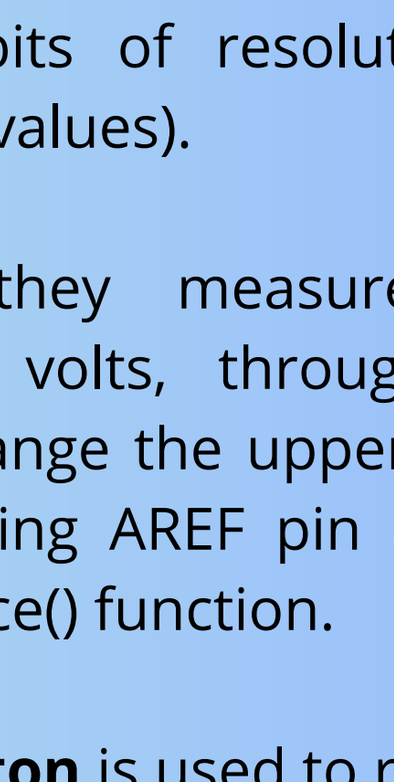
PWM signals are used for speed control of DC motors, dimming LEDs and many more.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provides 10 bits of resolution (i.e 1024 different values).

By default, they measure from ground to 5 volts, through is it possible to change the upper end of their range using AREF pin and the analogReference() function.

The **Reset button** is used to reset the Arduino Uno.





Pulse Width Modulation (PWM)

Aduino Uno board can be powered via the USB connection or with an external power supply which is barrel jack and Vin pin (battery).

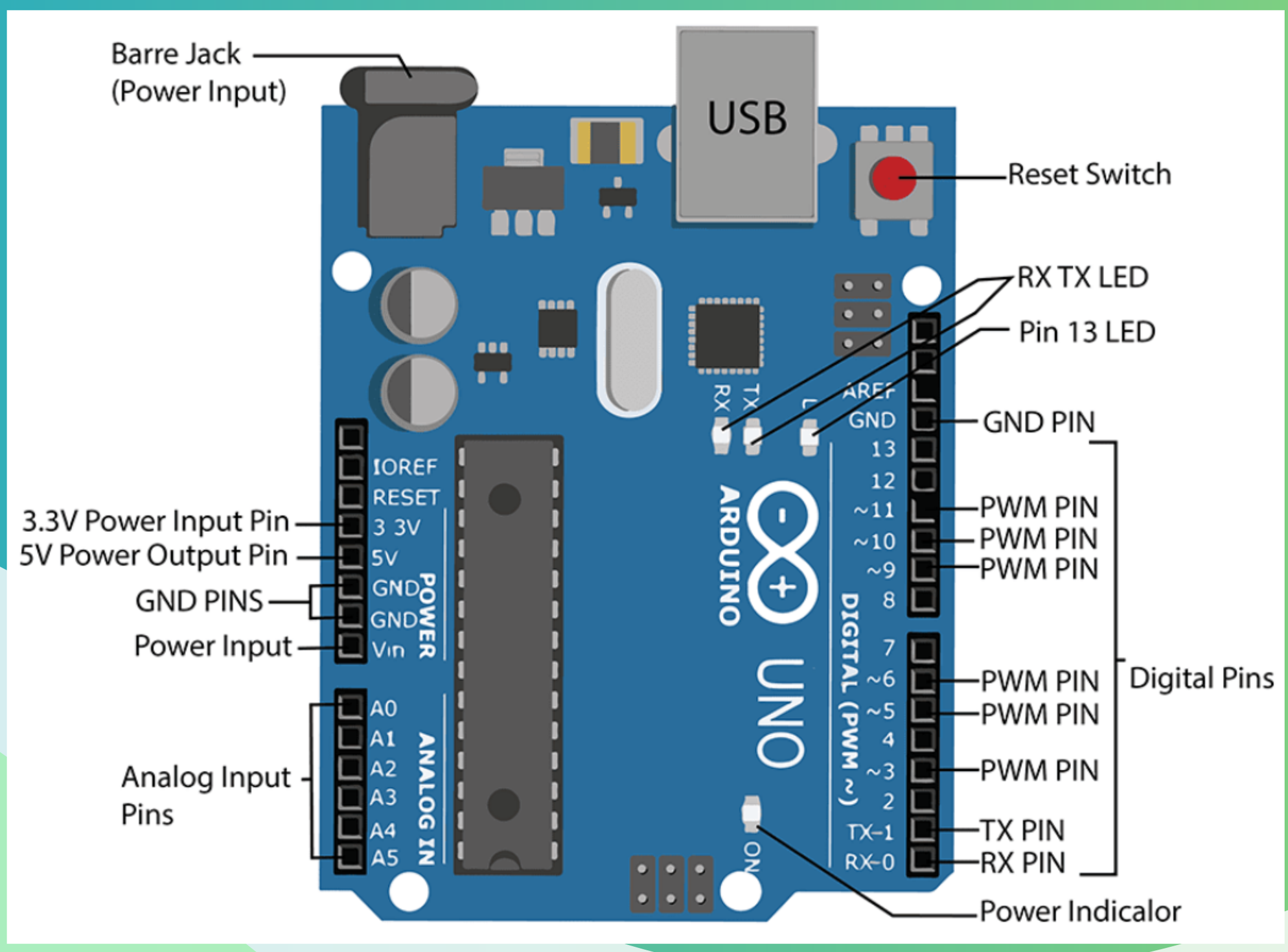
The board can be powered by 5-20 volts but the manufacturer recommends to keep it between 7-12 volts. If using more that 12V, the voltage regulator may overheat and damage the board and below 7 volts, might not suffice.

Three ways to power the Arduino:

- **USB Cable** - The USB cable is connected to the computer and will be powered at 5 volts with 500mA.
- **Barrel Jack** - The barrel jack or DC power jack can be used to power Arduino board. The barrel jack usually connected to AC-DC adapter.
- **Vin Pin** - This pin is used to power the Arduino Uno board using an external power source.

The power pins on the Arduino board are as follows:

- **GND Pin**- In the Arduiono Uno pinout, there are 5 GND pins or Ground pins, which are all interconnected
- **5V Pin**- This pin outputs a regulated 5V from the regulator on the board to power external components. The board can be supplied with power either from DC power jack (7-12V), the USB connector (5V) or the Vin pin of the board (7-12V).
- **3.3V Pin** - A 3.3V supply generated by the on-board regulator to power external components. Maximum current draw is 50mA.



Arduino Uno Pinout

TOPIC

B

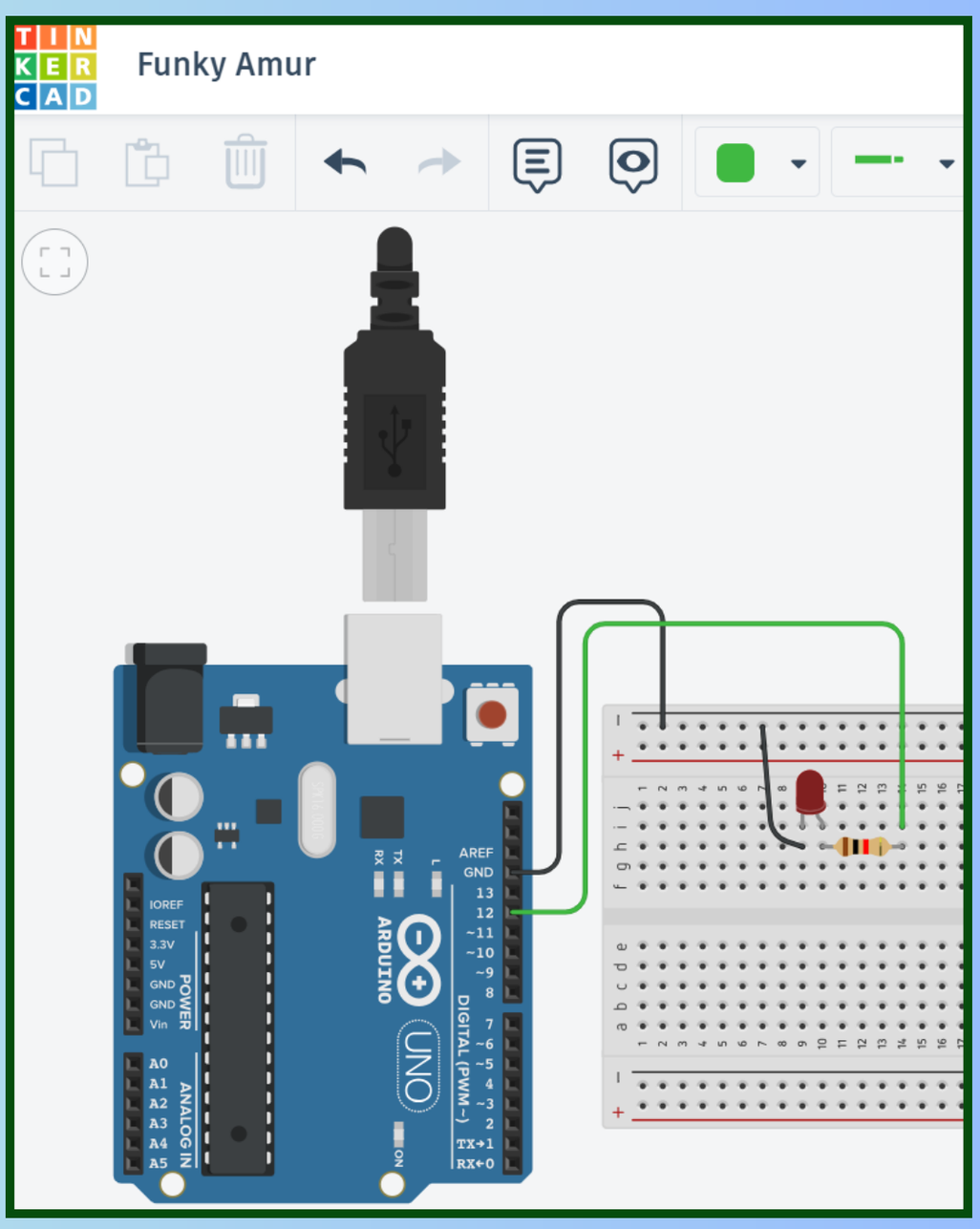
TINKERCAD

Tinkercad is a free of charge, easy to use apps for 3D design, electronics and coding that can runs in a web browser.

Tinkercad's circuits section is a browser-based electronic circuit simulator that supports Arduino Uno microcontrollers.

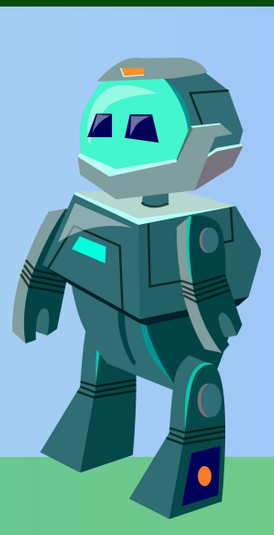
Code for Arduino Uno project can be created using **blocks-based code** or **text-based code.**

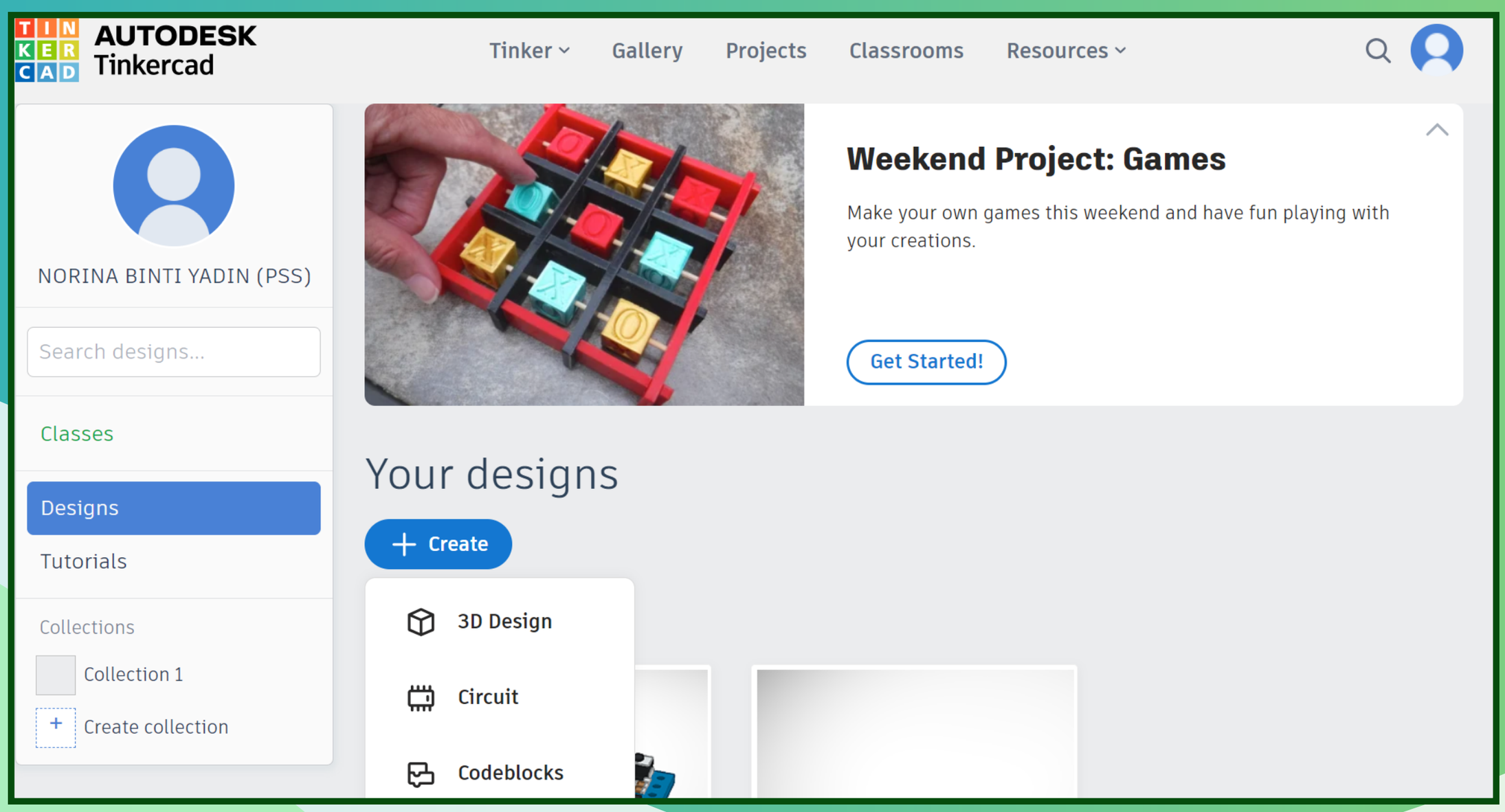
Tinkercad comes with built-in libraries for popular components.



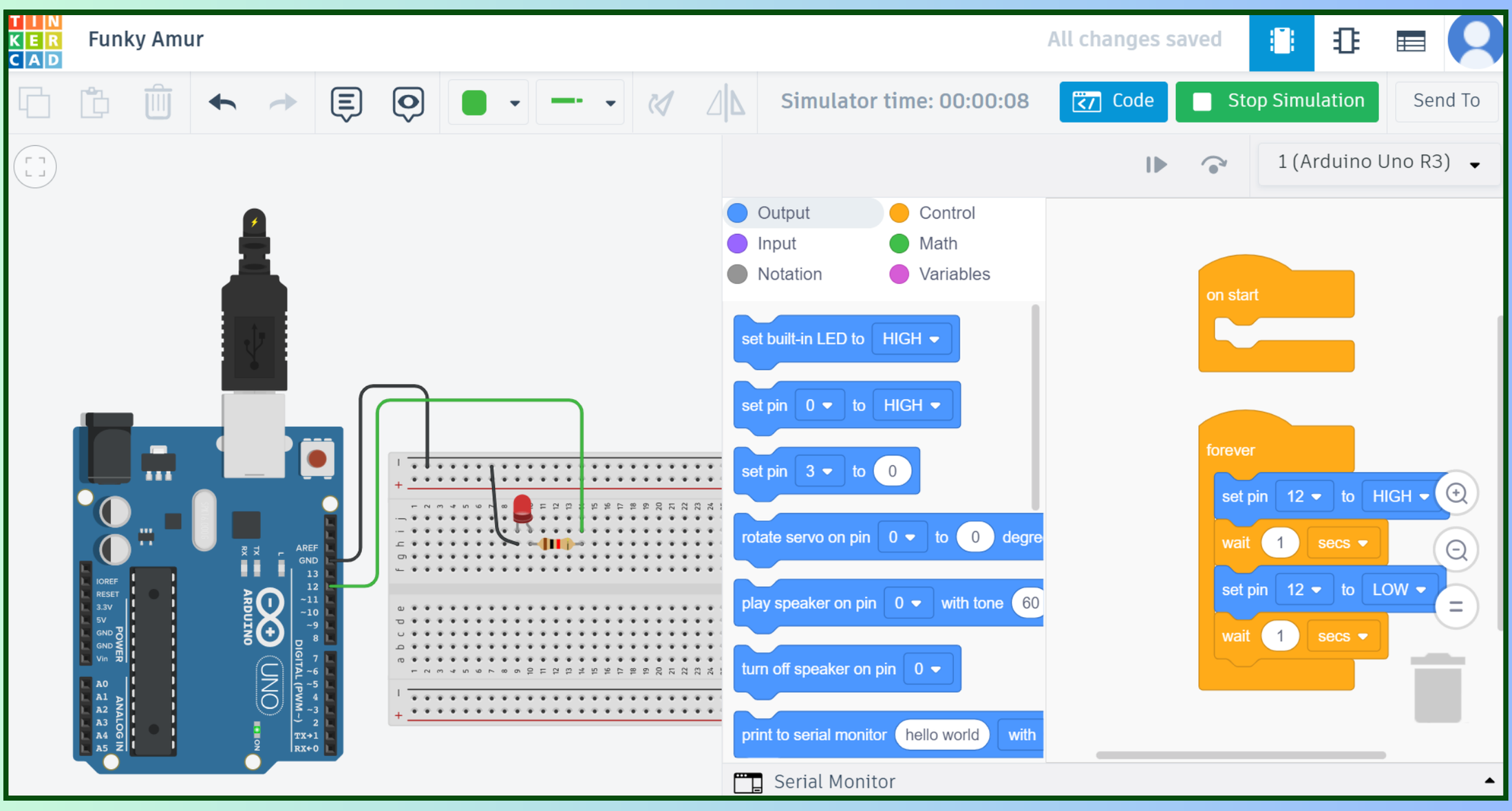
TINKERCAD

AUTODESK®
TINKERCAD®

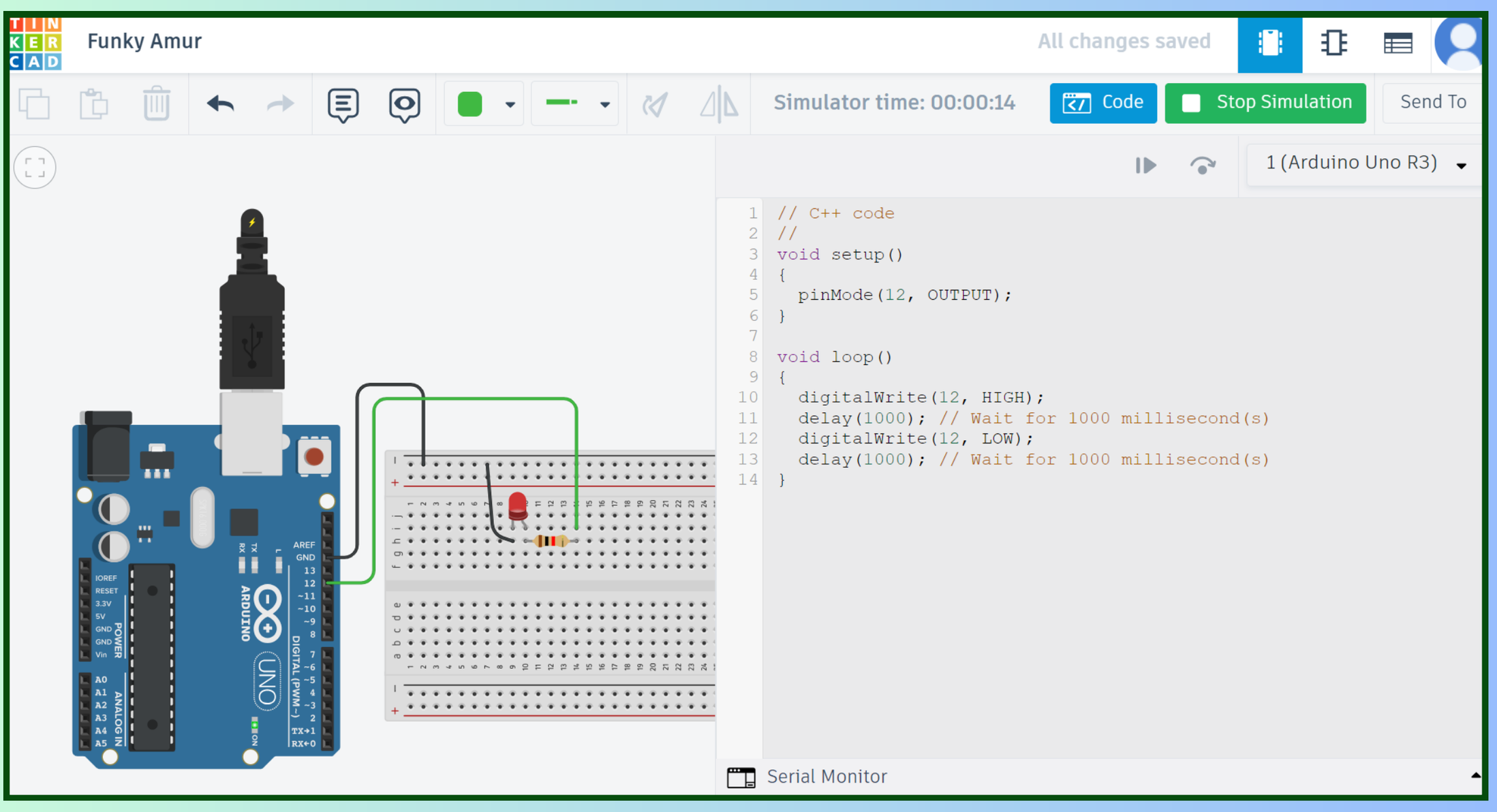




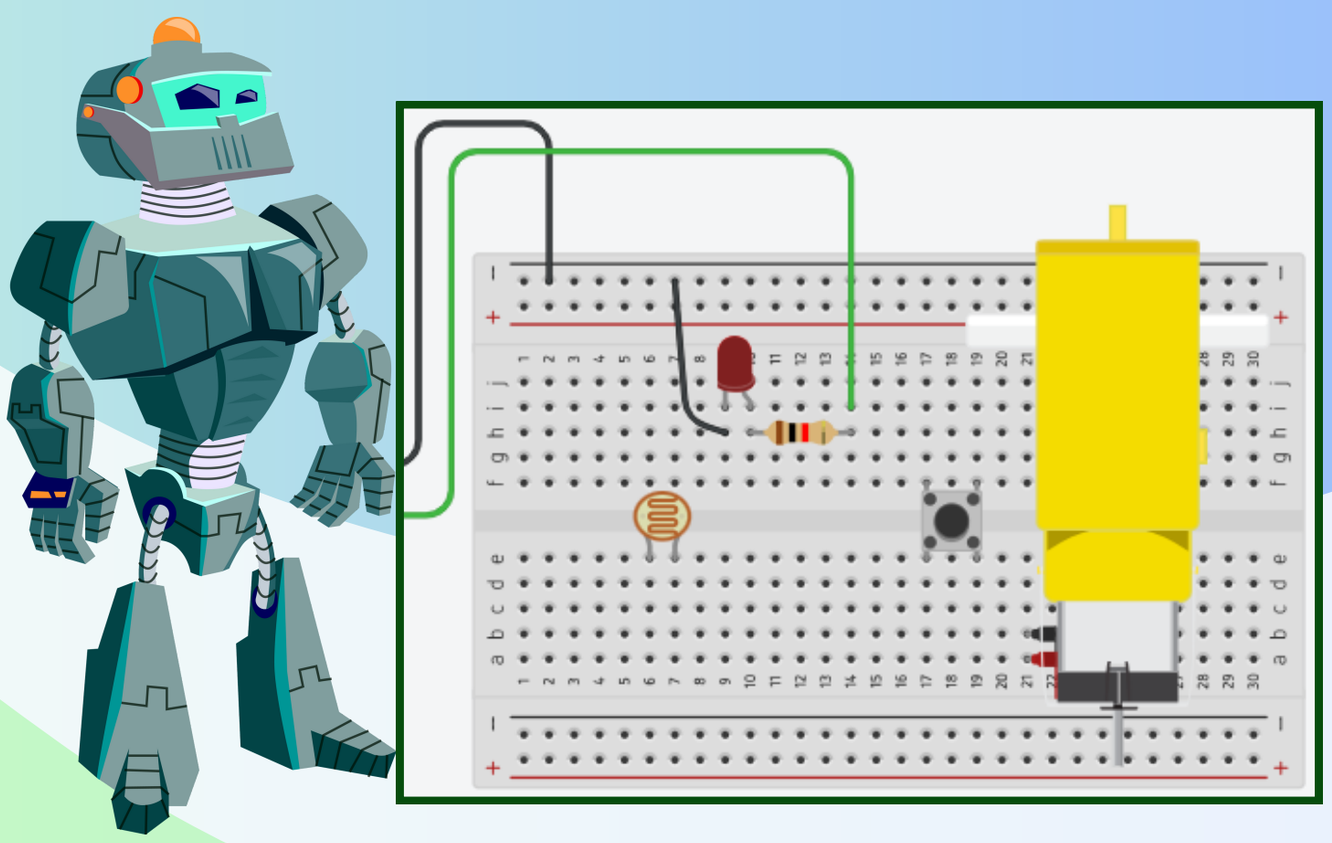
Tinkercad Web Browser



Blocks-based Code



Text-based Code



TOPIC

C

COMPONENTS

Breadboard

Breadboard is a construction base used to build semi-permanent prototypes of electronic circuits.

Breadboards do not require soldering or destruction of track and are hence reusable.

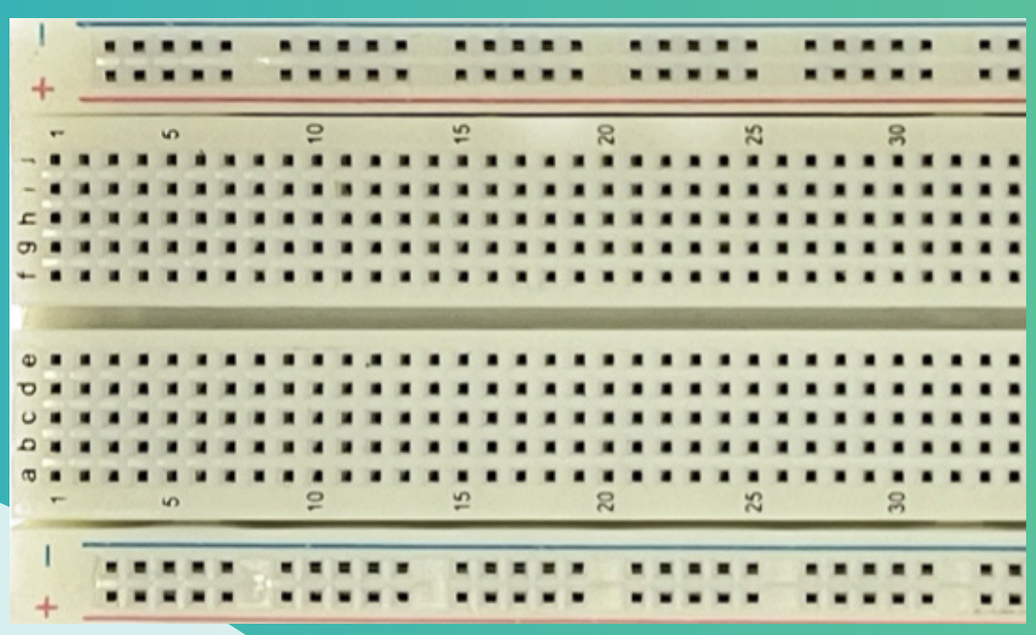
Breadboards connect pin to pin (between its hole by metal strips inside the breadboard. Some vertical connections and some horizontal connections.

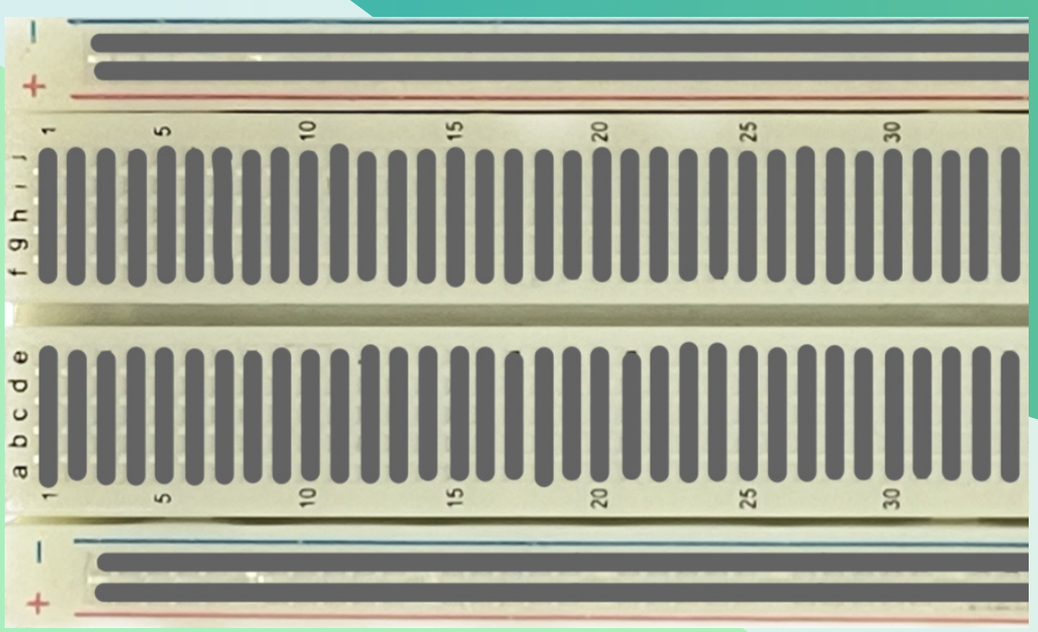
Breadboard is made up of two types of areas called **terminal strips** and **bus strips.**

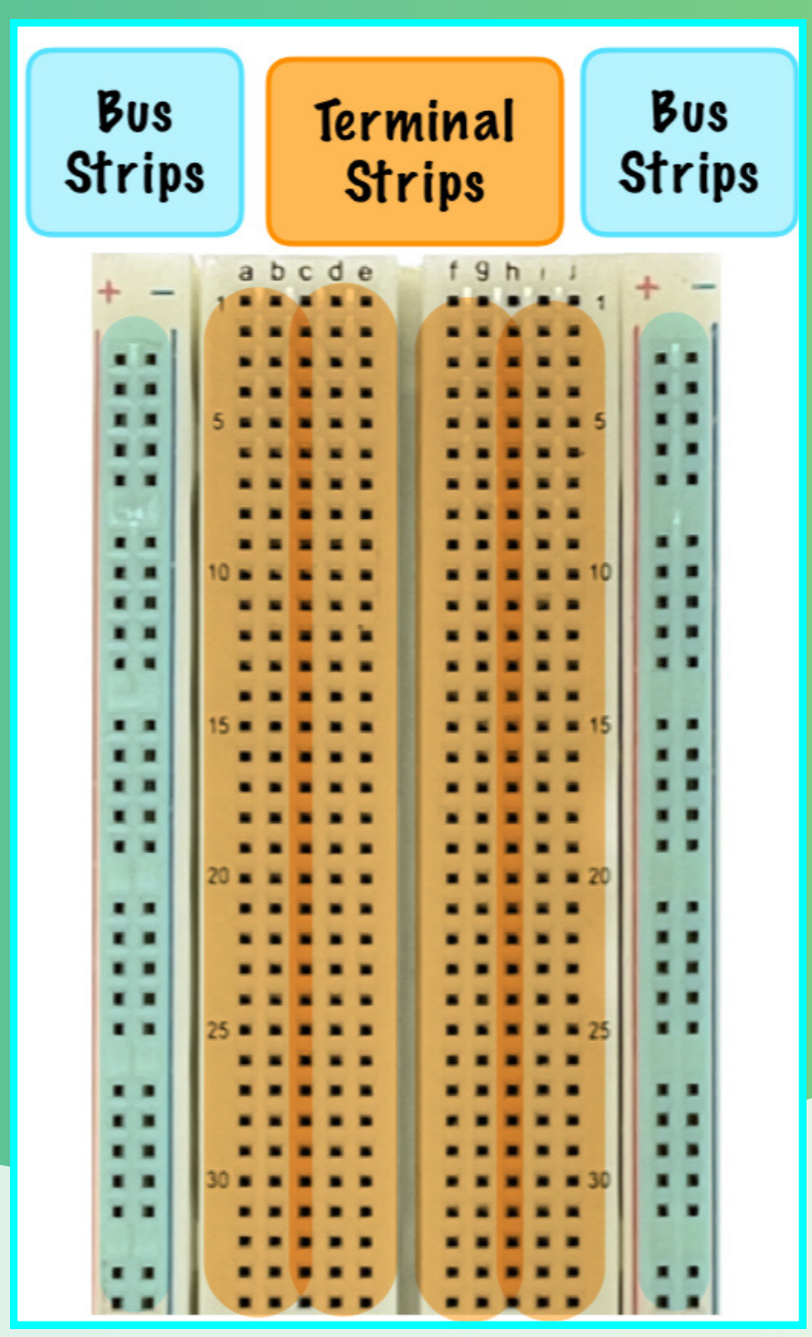
Terminal strips are the area that hold most of the electronic components. In the terminal strip of breadboard, the holes are connected horizontally, in rows. The left side is separated from the right side.

Five clips in a row on each side are electrically connected. The five columns on the left often marked as A, B, C, D and E, while the ones on the right are marked F, G, H, I and J.

To provide power to the electronic components, bus strips are used. A bus strip usually contains two columns. One column for ground (GND) and one column for a supply voltage.







Jumper Wires

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering.

Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed.

Jumper wires colors can be used to differentiate between types of connections, such as ground or power.

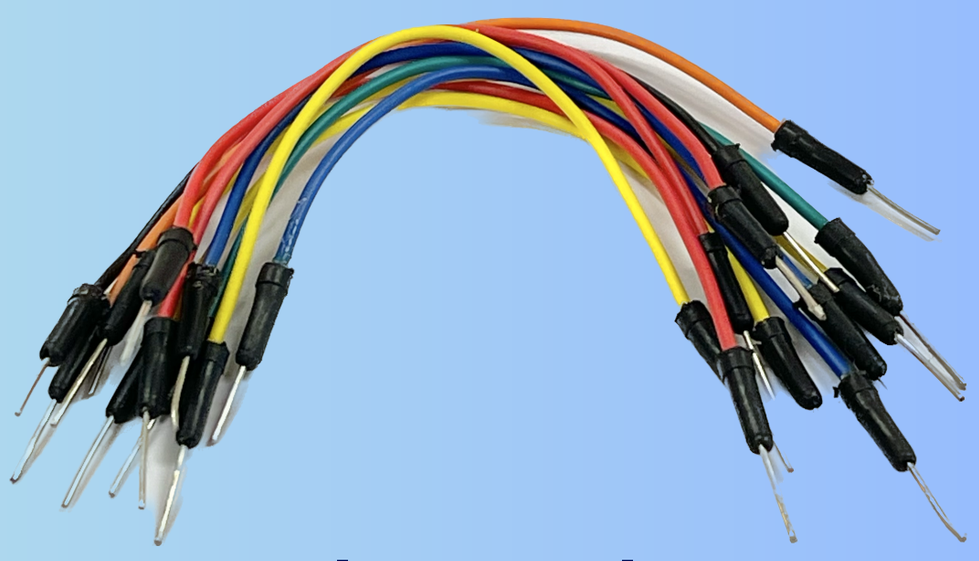
Jumper wires typically come in three versions:

- **Male-to-male**
- **Male-to-female**
- **Female-to-female.**

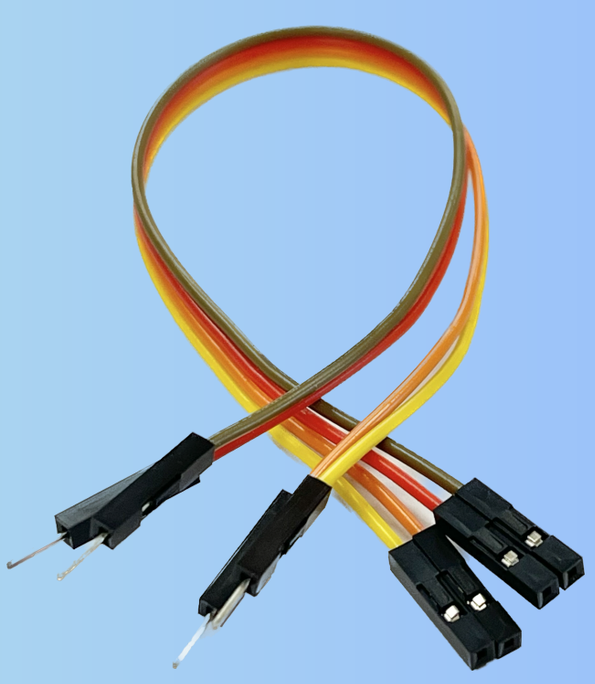
The difference between each is in the end point of the wire. Male ends have a pin protruding and can plug into things, while female ends do not but are also used for plugging.

Male connector is referred to as a plug and has a solid pins for centre conduction.

Meanwhile, a female connector is referred to as a jack and has a centre conductor with a hole in it to accept the male pin.



Male-to-male



Male-to-female

Female-to-female

Resistor

The **Resistor** is an electrical component that limits or regulates the flow of electrical current in an electronic circuit.

Resistors are passive components, meaning they only consume power and cannot generate it.

Commonly, resistors are used to limit current, divide voltages and pull-ip I/O lines.

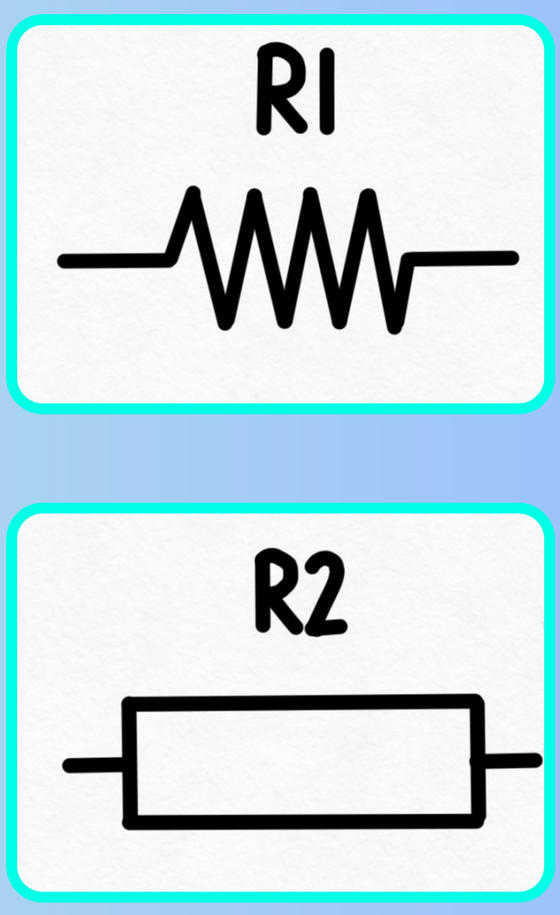
The electrical resistance of a resistor is measured in ohms (Ω).

Resistors have two terminals, one connection on each end of the resistor.

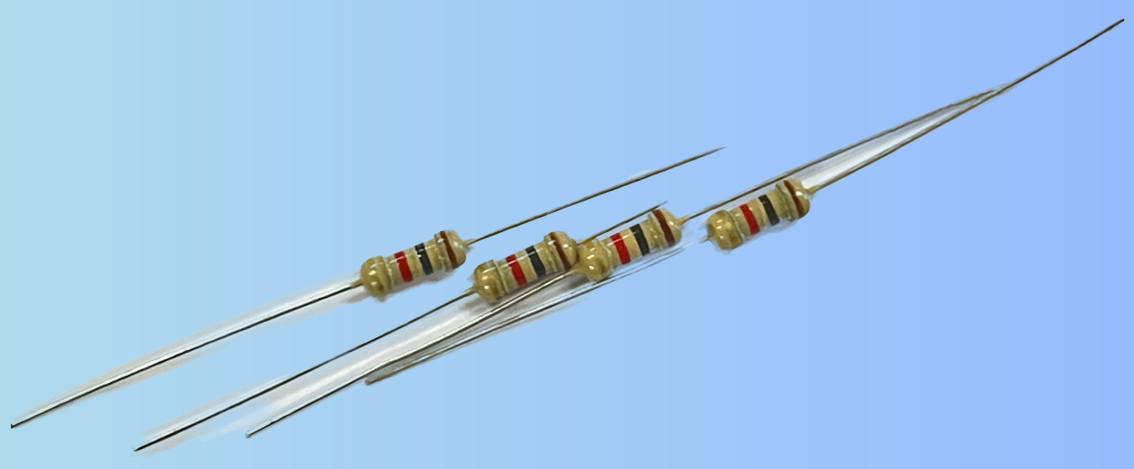
In this book, we used three types of resistor with different ohm (Ω) as follows:

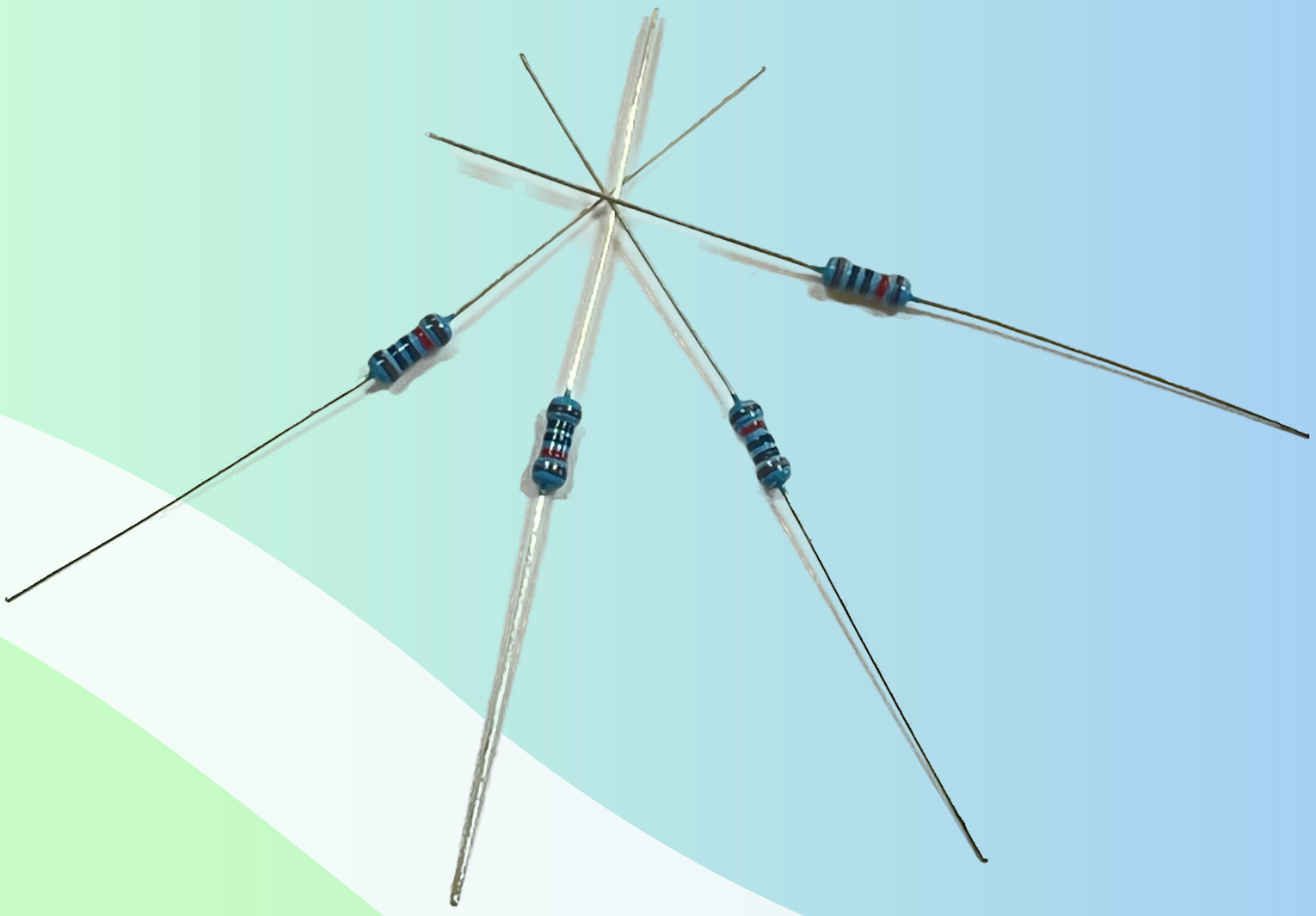
- 330 Ω
- 1000 Ω
- 10K Ω

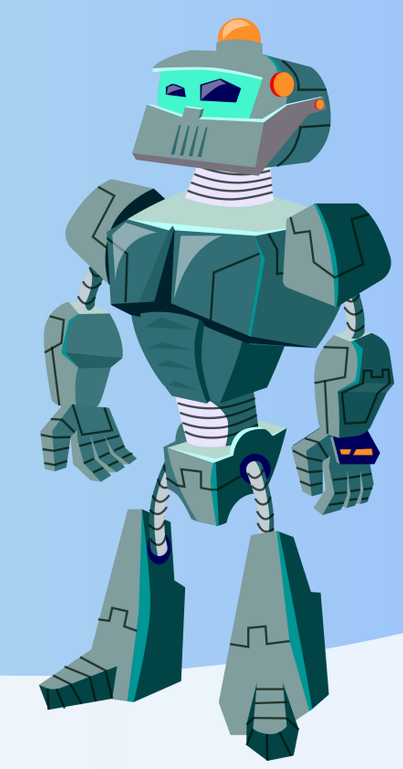
When modeled on a schematic, a resistor will show up a s one of these two symbols:



Symbols of Resistor







Light Emitting Diode (LED)

LED is a widely used standard source of light in electrical equipment. LED is a semiconductor device that emits light when an electric current flows through it.

LED allow electrical current to flow in one direction, therefore LEDs are polarised.

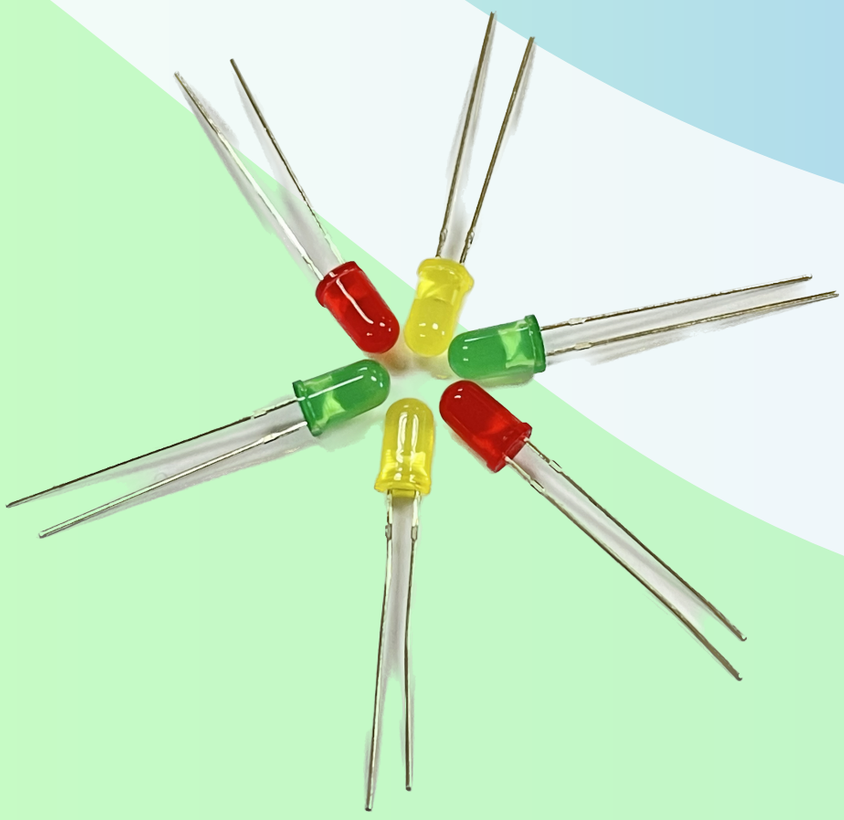
With LEDs, these positive leads are referred to as **anodes** and negative leads are referred to as **cathodes.**

Electricity enters through the positive side (anode) and exists through the negative side (cathode).

The anode lead is always connected to the positive terminal of the battery and the cathode lead is connected to the negative terminal of the battery.

There are several color light of LEDs used in these book such below:

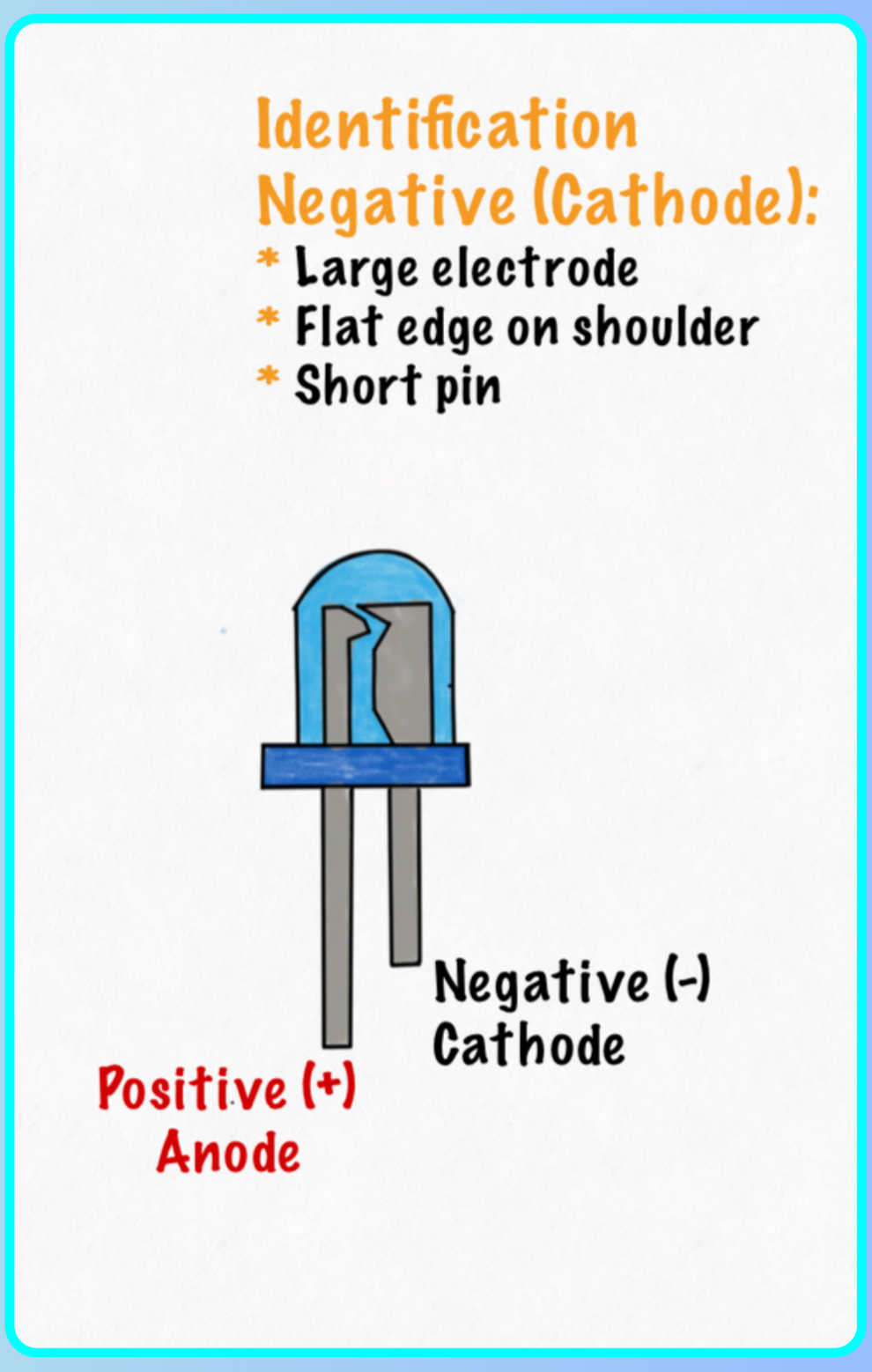
- Yellow LED
- Red LED
- Green LED



It is very important that LEDs are connected to a circuit in the right direction.

If the LED is connected in the wrong direction it will not illuminate and may damage the LED.

The longer pin indicates that it's the anode while the shorter pin is the cathode as shown in the figure below.



LED Polarity

RGB LED

RGB LED is a combination of three (3) LEDs in just one package:

- **R**ed LED
- **G**reen LED
- **B**lue LED

Basically, RGB LEDs are categorize as either common anode or common cathode.

RGB LEDs have four leads. The longest lead is second from the left, the leads should be in the following order: red, anode (+) or cathode (-), green and blue.

For common anode, apply a ground to the red, green and blue leads and connect the anode lead to the positive terminal of the power supply.

For common cathode, apply a VCC to the red, green and blue leads and connect the cathode lead to the negative terminal of the power supply.

RGB LED can produce almost any color, such as color code below by combining red, green dan blue colors.

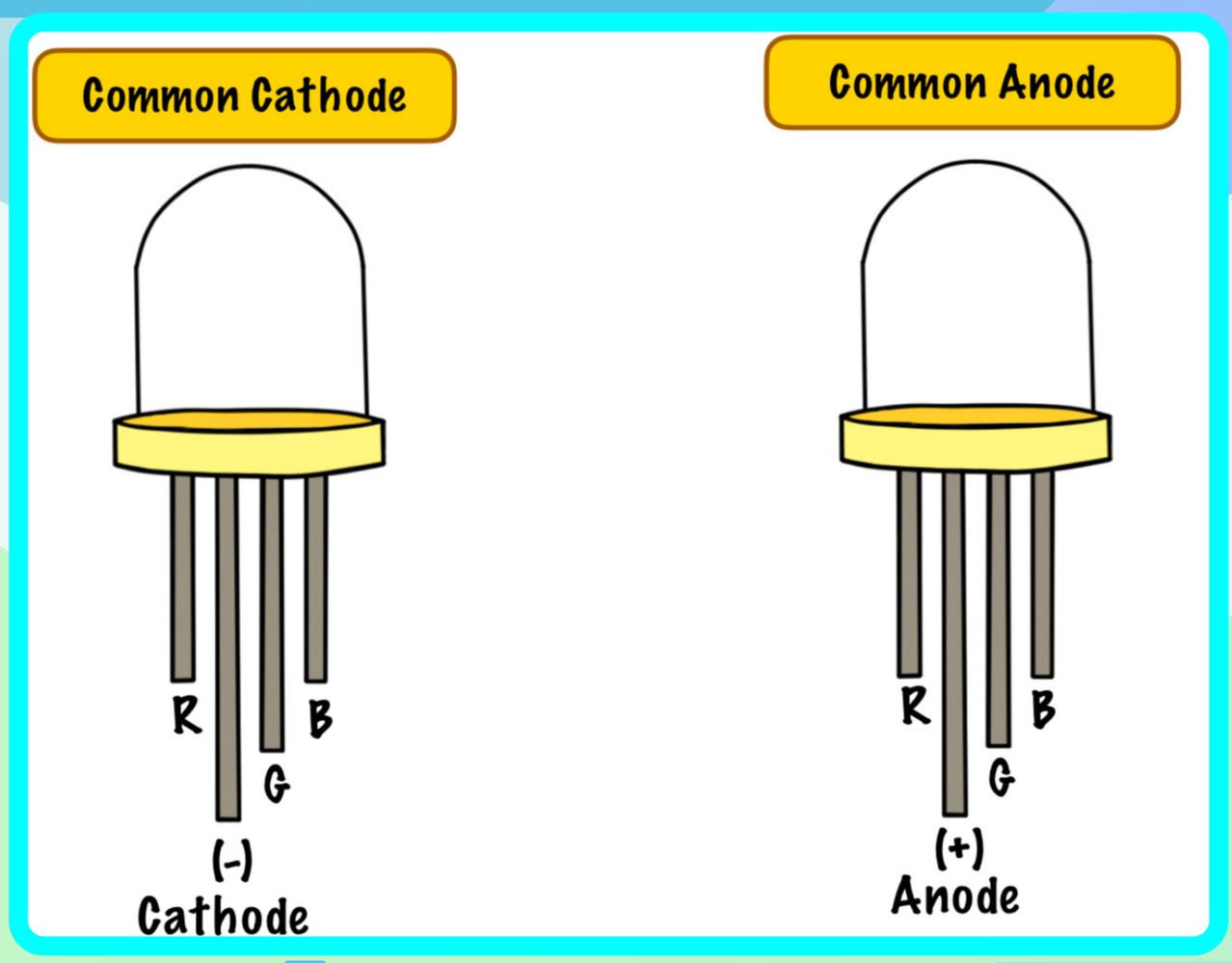
For example, to produce Magenta light, we should set the red LED to 255, green LED set as 0 and blue LED set as 255.

For white light, we should set all three LEDs to the highest intensity which is 255.

Color	HTML/CSS NAME	Decimal (R,G,B)
	Black	(0,0,0)
	White	(255,255,255)
	Red	(255,0,0)
	Lime	(0,255,0)
	Blue	(0,0,255)
	Yellow	(255,255,0)
	Cyan	(0,255,255)
	Magenta	(255,0,255)
	Silver	(192,192,192)
	Grey	(128,128,128)
	Maroon	(128,0,0)
	Olive	(128,128,0)
	Green	(0,128,0)
	Purple	(128,0,128)
	Teal	(0,128,128)
	Navy	(0,0,128)

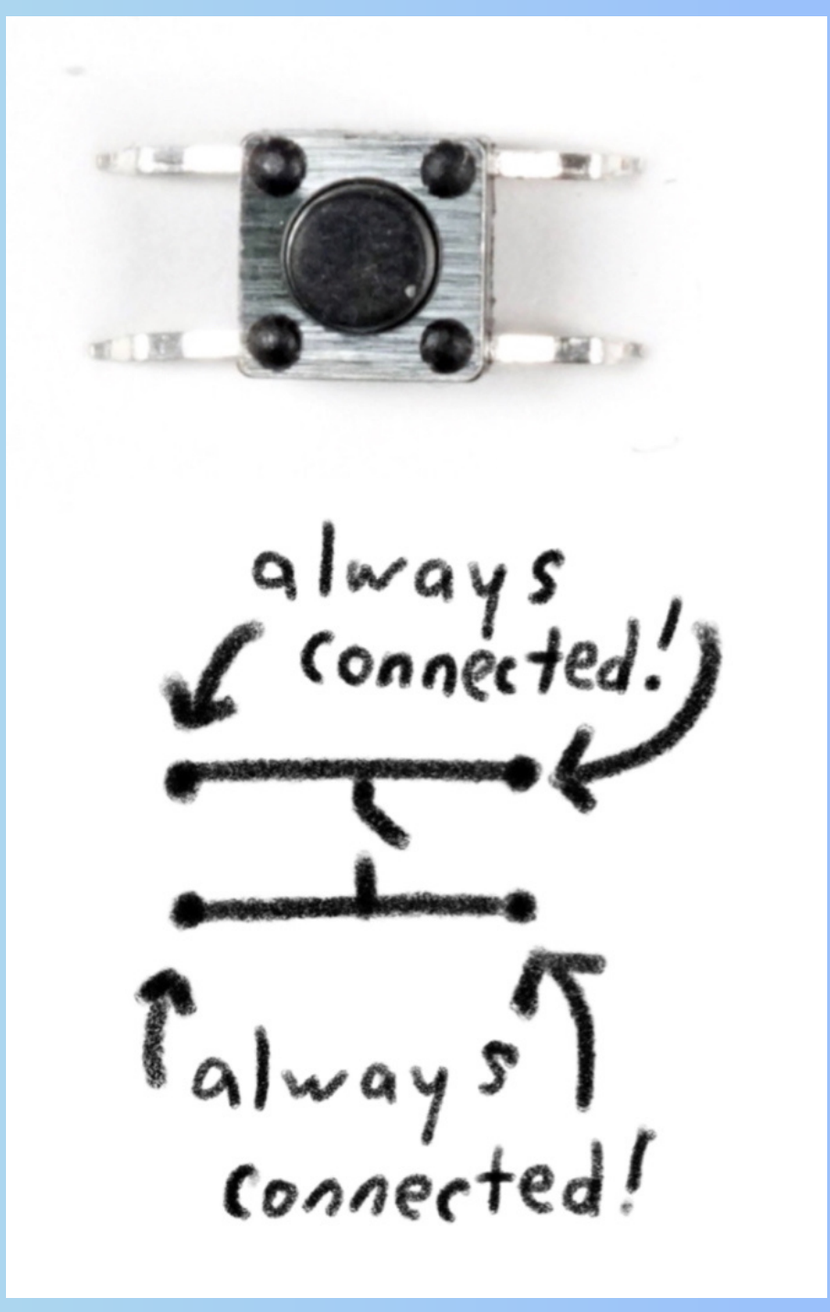
Color Code of RGB LED

RGB LED Polarity



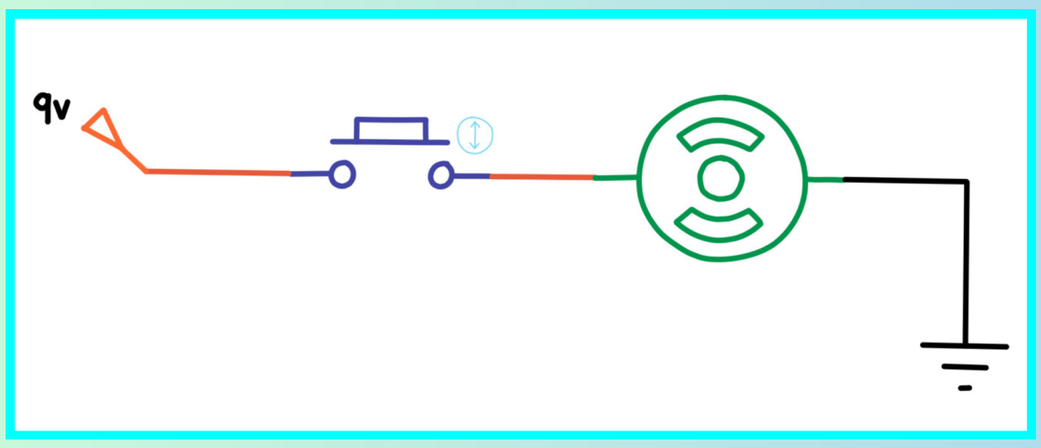
Push Button Switch

Push Button Switches are normally-open tactile switches. The push button allows us to power the circuit or make any particular connection only when we press the button.



This four-legged little button makes a nice "click” sound when we push it. It makes the circuit connected when pressed and breaks when released.

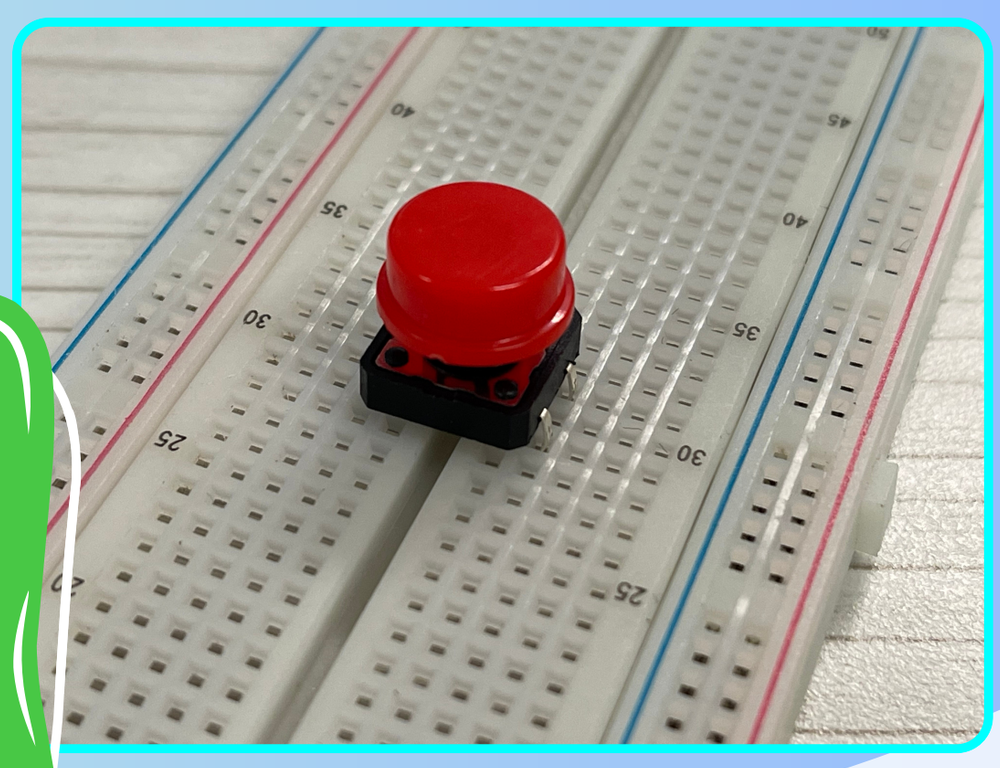
When connecting it between the supply and the circuit, we should only connect the wires with both the legs of the Push-Button as shown in the circuit below:



Push-Button Connection

Note that the upper right and upper left legs are always connected to each other, and the lower left and lower right legs are always connected to each other.

The action of the button is to connect the upper part of the switch to the lower part of the switch.



The button is designed with spacing specifically meant to fit across the center dividing valley in a standard breadboard.

Light Dependent Resistor (LDR)

Light Dependent Resistor (LDR) as the name states is a special type of resistor that works on the photoconductivity principle means that resistance changes according to the intensity of light.

LDR is also known as photoresistor, photocell, photoconductive cell or light sensor.

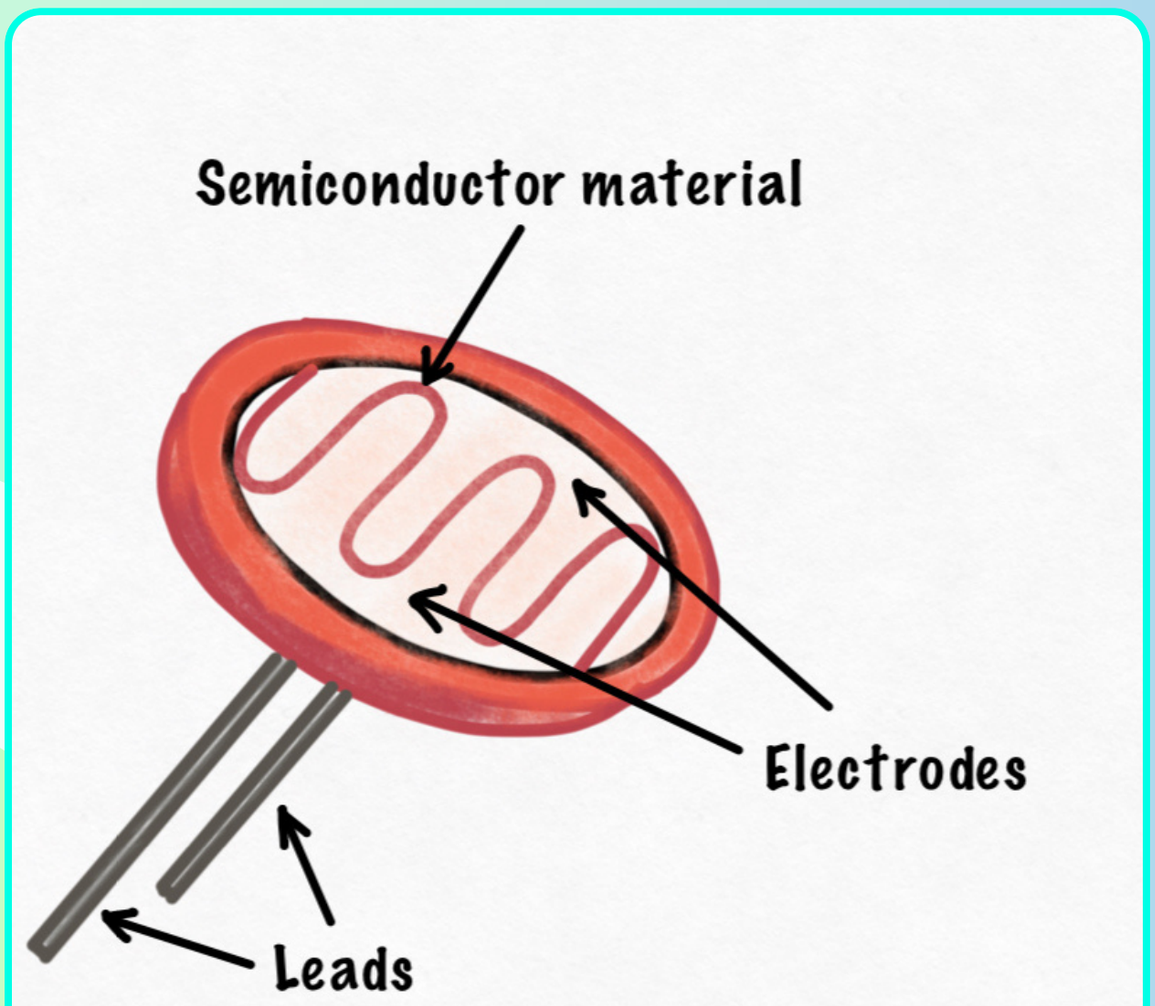
LDR is usually available in 5mm, 8mm, 12mm and 25mm dimensions.

It is often used as a light sensor, light meter, automatic streetlight and in areas where we need to have light sensitivity.

LDR are passive components, and working principle of this component is photoconductivity.

The specification of LDR is as follows:

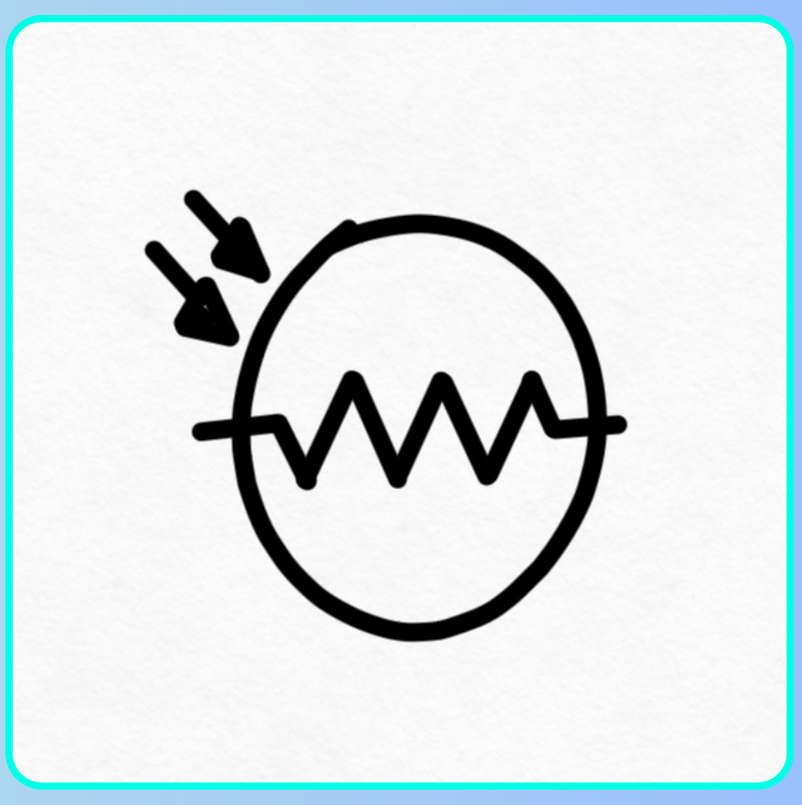
- Input voltage: DC 3.3V to 5V
- Output: Analog and Digital
- Sensitivity adjustable



LDR PinoutPush-Button Connection

The LDR has the highest resistance in dark, around 1012 Ohm and this resistance decreases with the increase in light.

When modeled on a schematic, LDR will show up with these symbols:



Symbols of LDR

KY-012 Active Piezo-Buzzer Module

KY-012 Active Piezo-Buzzer Module creates an audible sound at 2.5 kHz without the need for pulse width modulation (PWM) or any additional complex code.

This electronic component only need an external power supplied to make a sound.

The buzzer was found to draw around 25mA from a 5V power supply.

Because the buzzer draws more current than the maximum of 20mA specified per Arduino [in, it is not a good idea to power it directly from an Arduino pin. It is better to use a transistor to power the buzzer.

The specifications of this electronic component as below:

- Operating voltage: 3.5V - 5.5V
- Maximum Current: 30mA / 5VDC
- Resonance Frequency: 2500 Hz ± 300 Hz
- Maximum Sound Output: 85 Db @ 10cm

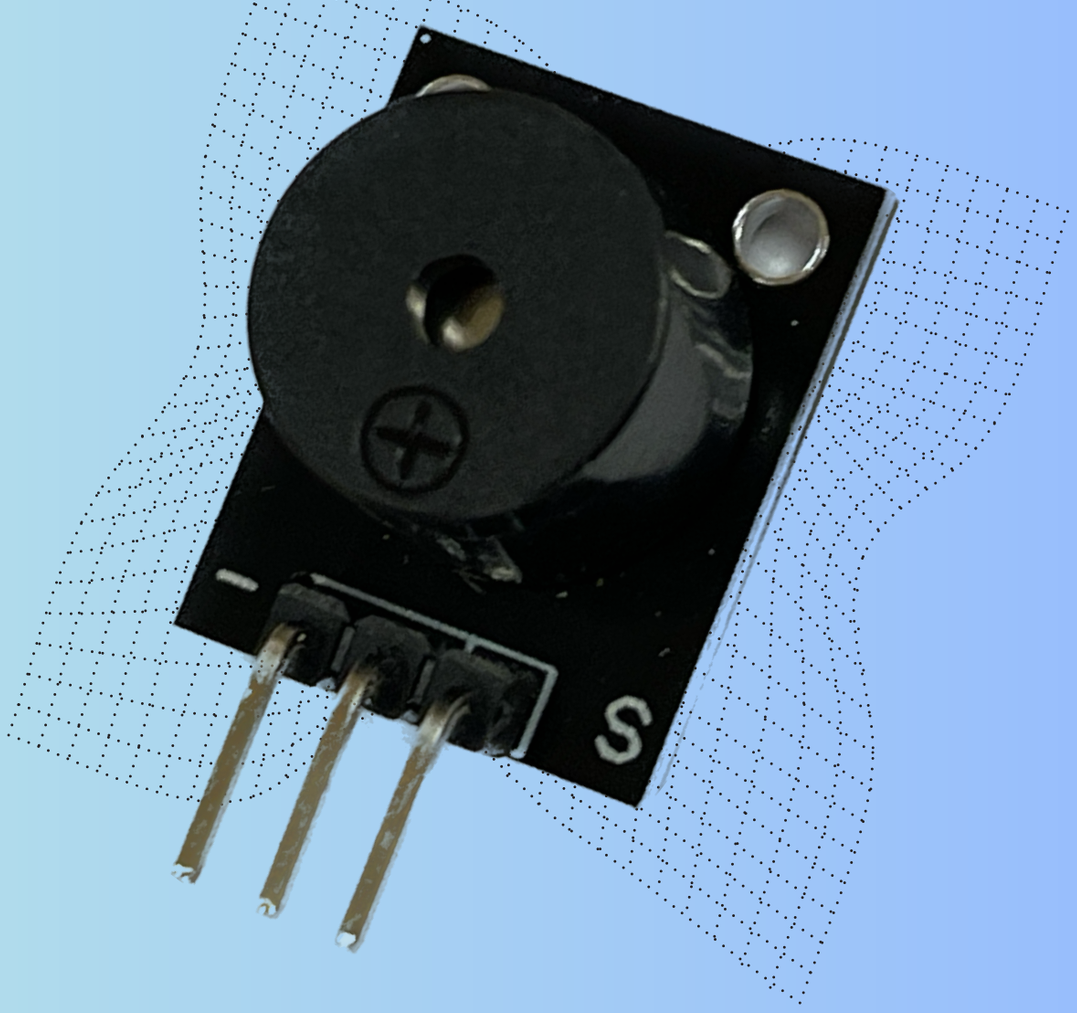
The buzzer module is active because it has internal electronics to generate a sound or tone.

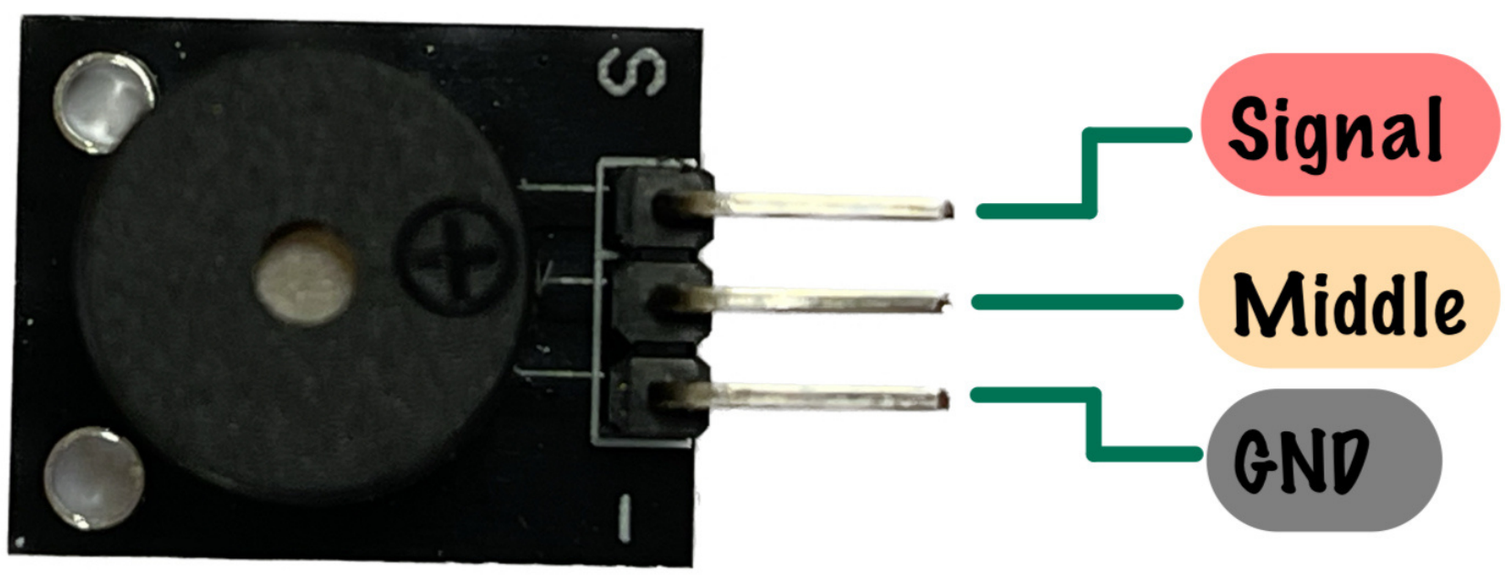
The volume of the buzzer can be changed by changing the supplied voltage.

The pitch of the tone is generated by the internal electronics and cannot be changed.

The active buzzer includes 3-pins:

- **Signal** pin: Used for power (3.3V-5V)
- **GND** pin: Ground (0V)
- **Middle** pin: Not connection





Piezo-Buzzer Module Pinout

Infrared (IR) Sensor

The Infrared (IR) sensor is an inexpensive solution to avoidance detection for robotics, smart cars, and other electronic uses.

The sensor has a very good and stable response even in ambient light or in complete darkness.

IR sensor has a built-in **IR transmitter** and **IR receiver** that sends out IR energy and looks for reflected IR energy to detect the presence of any obstacle in front of the sensor module.

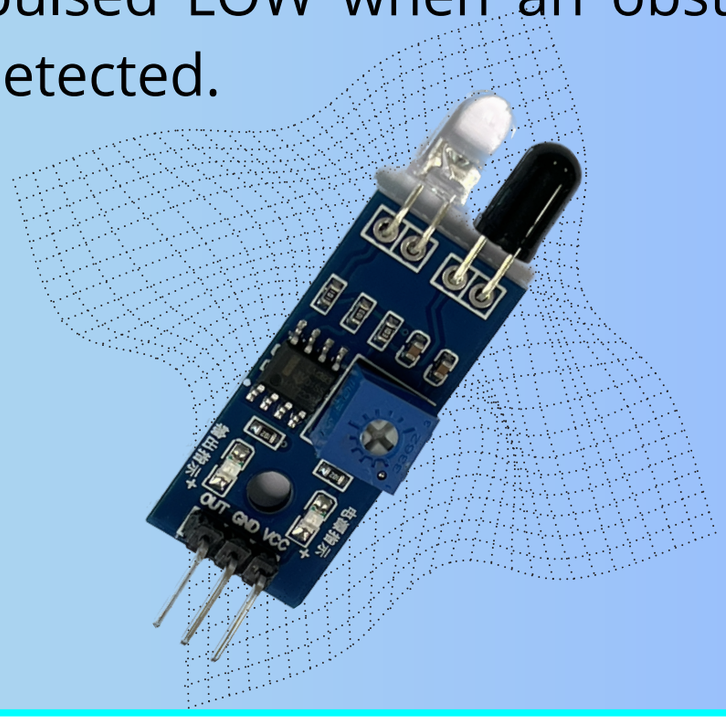
The IR transmitter (clear LED) emits the IR signal. The IR receiver (black LED) looks for the reflected IR signal to determine if the object is present or not. The presence of obstacle is reflected in the OUT pin.

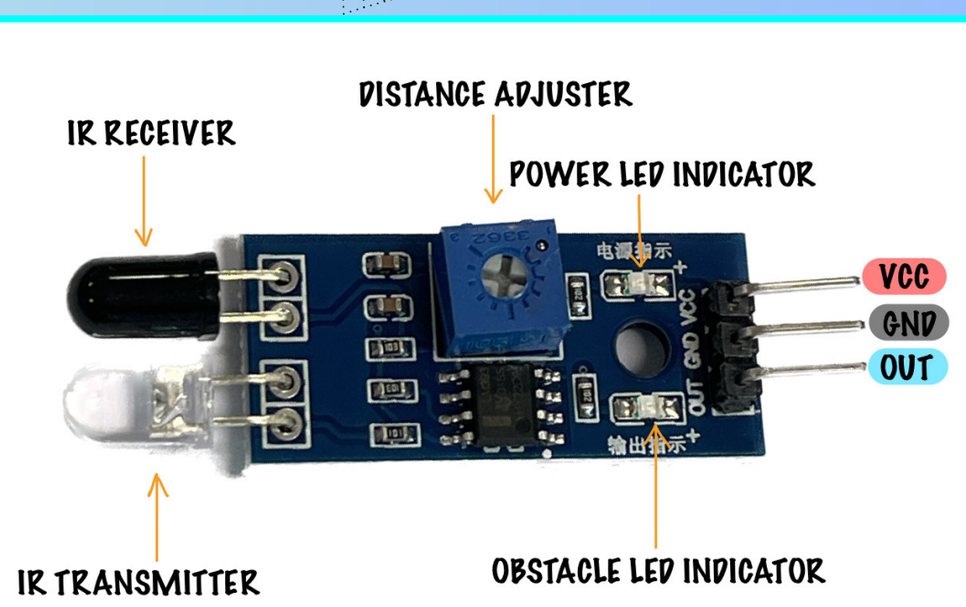
- If the obstacle is present in front of the sensor, the sensor's OUT pin is LOW.
- If the obstacle is not present in front of the sensor, the sensor's OUT pin is HIGT.

The module has an onboard potentiometer that lets users adjust the detection range between 2~30cm. The working voltage is from 3.3V to 5V.

The ultrasonic sensor includes 3-pins:

- **VCC** pin: Used for power (3.3V/5V)
- **GND** pin: Ground (0V)
- **OUT** pin: Normally HIGH. Output is pulsed LOW when an obstacle is detected.





IR Sensor Pinout

Ultrasonic Sensor

Ultrasonic sensors are used as proximity sensors. Ultrasonic sensors have two main components: the **transmitter** and the **receiver**.

The ultrasonic Sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves and converts the reflected sound into an electrical signal.

The distance is determined by measuring the travel time of ultrasonic sound and its speed.

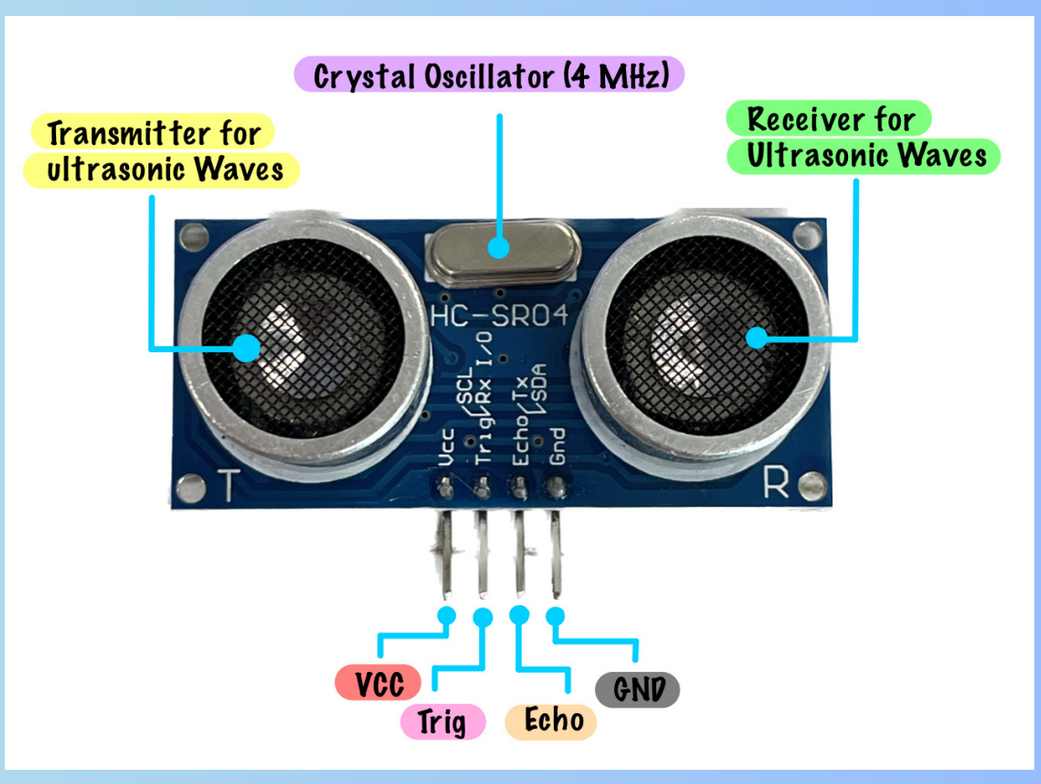
$$\text{Distance} = \frac{Time \times Speed\ of\ Sound}{2}$$

The speed of sound in air is approximately 343 meters per second (m/s) or 0.0343 centimeters per microsecond (cm/ms).

Ultrasonic sensors are an excellent way to measure the distance, speed and position of object. Ultrasonic sensors are also used as level sensors to detect, monitor, and regulate liquid levels in closed containers.

The ultrasonic sensor includes 4-pins:

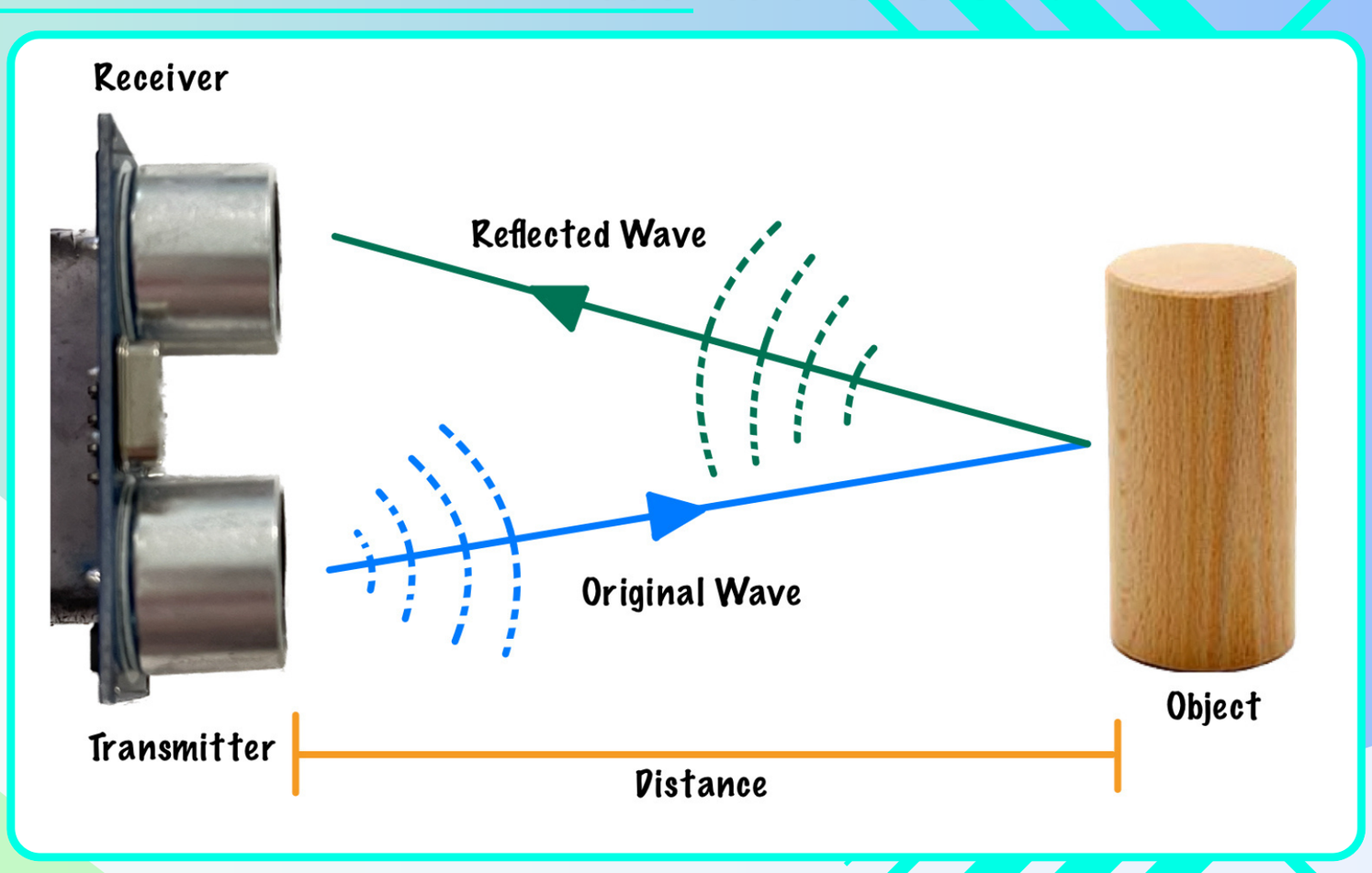
- **VCC** pin: Used for power (5V)
- **GND** pin: Ground (0V)
- **Trig** pin: Input pin
- **Echo** pin: Output pin



Ultrasonic Sensor Pinout

To generate the ultrasonic sound, the trigger pin needs to be triggered as HIGH. The trig pin will send the ultrasonic wave from the transmitter. It will be received by the receiver called Echo pin and it will calculate the output time to measure the distance.

Ultrasonic sensors will send sound waves in the target direction and calculate its distance by timing how long it takes for the waves to bounce back to the sensor.



Soil Moisture Sensor

Soil Moisture Sensor is a type of sensor used to measure the volumetric water content of the soil.

It measures the volumetric content of water inside the soil and gives us the moisture level as output.

The moisture sensor consists of two probes that are used to detect the moisture of the soil.

These two probes are used to pass the current through the soil and then the sensor reads the resistance to get the moisture values.

The specification of this sensor includes the following:

- Required voltage range for working is 3.3V to 5V
- Required current for workinh is <20mA
- Tupe of interface is analog

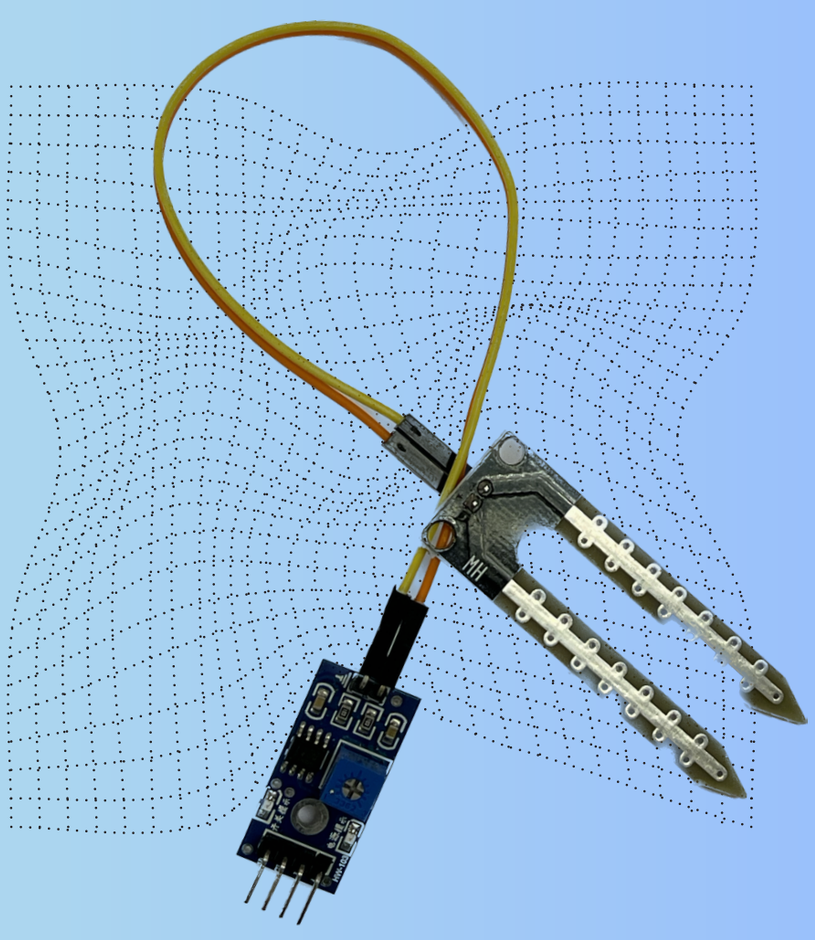
This soil moisture sensor is integrated with the LM393 comparator chip, this is an integrated circuit (IC).

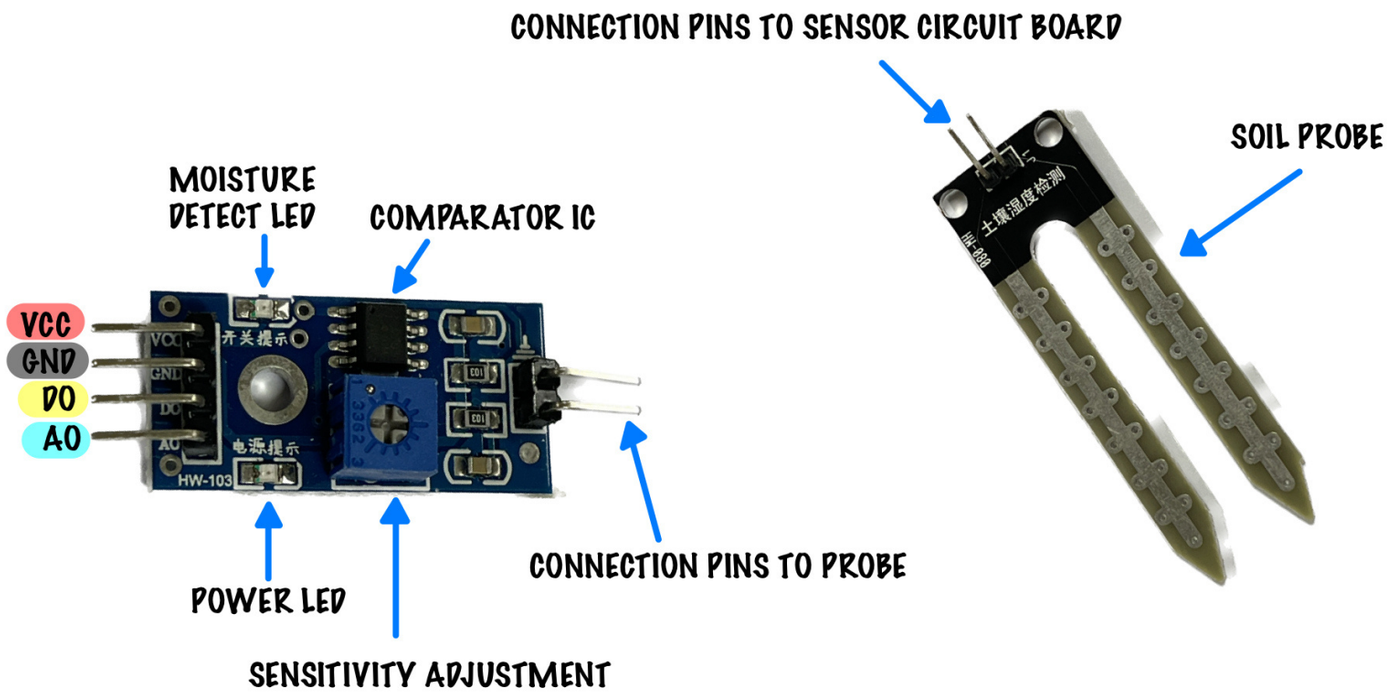
They are commonly used in devices that measure analog signals and act as Analog-to-digital converter (ADC).

This module has built-in potentiometer to set the sensitivity of the digital output and adjust the threshold level.

The FLM393 Comparator integrated with soil moisture sensor includes 4-pins:

- **VCC** pin: Used for power (3.3V/5V)
- **A0** pin: Analog output
- **D0** pin: Digital output
- **GND** pin: Ground (0V)





Soil Moisture Sensor Pinout

HC-06 Bluetooth Module

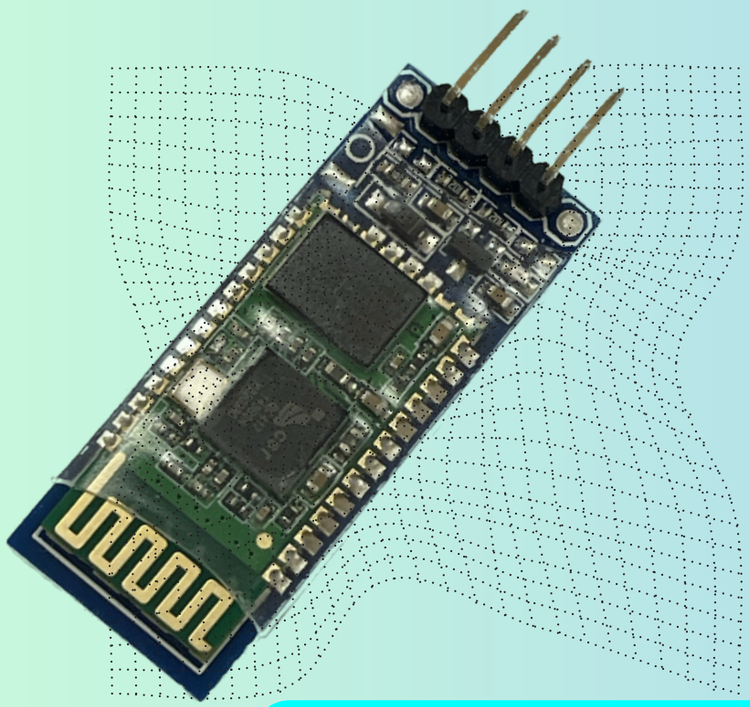
HC-06 Bluetooth Module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. Hence, there is no cable required for sending and receiving data for this module.

HC-06 is a Bluetooth Module that provides serial communication (UART), designed to facilitate wireless communication between microcontrollers and other Bluetooth-enabled devices.

Its communication is via serial communications which makes it an easy way to interface with a controller, PC smartphone or tablet.

The specification of this Bluetooth Module includes the following:

- Typical - 80dBm sensitivity
- Operating current: 40MA
- Power level: Class 2 (+6dBm)



The advantage of this standard is that we can send data in fewer time intervals. It can effortlessly send the data in 0.5 seconds of an interval.

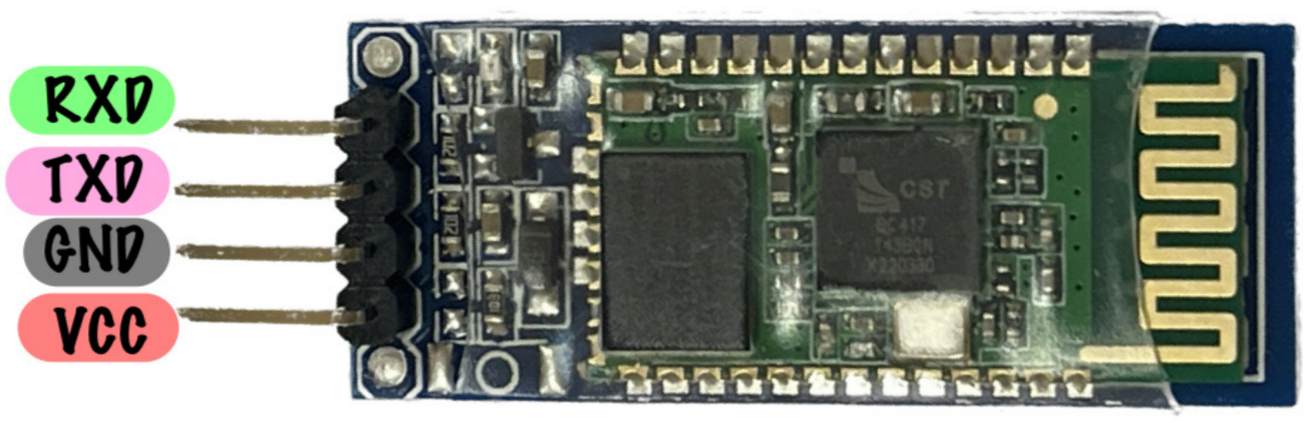
With this feature, the workload on Bluetooth chips can be reduced and we can send a large amount of data in a short time.

The HC-06 is the best option when short-distance wireless communication is needed. The module is used for ireless communications of less than 100 meters.

The module is very easy to interface with and communicate with. This module consumes very little power to function and can be used on battery-operated mobile systems.

The HC-06 Bluetooth Module includes 4-pins:

- **VCC** pin: Used for power (5V)
- **GND** pin: Ground
- **TXD** pin: Transmitter (Transmits Serial Data). Connect this pin with the RXD pin of the Microcontroller.
- **RXD** pin: Receiver (Receive Serial Data). Connect this pin with the TXD pin of the Microcontroller.



Bluetooth Module Pinout

L298N Motor Driver

L298N Motor Driver can control the speed and spinning direction of two DC motors by combining these two techniques:

- **PWM** - to control the speed
- **H-Bridge** - to control the spinning direction

At the center of the module is a big, black chip called L298N. The L298N chip contains two standard H-bridges capable of driving a pair of DC motors, making it ideal for building a two-wheeled robotic platform.

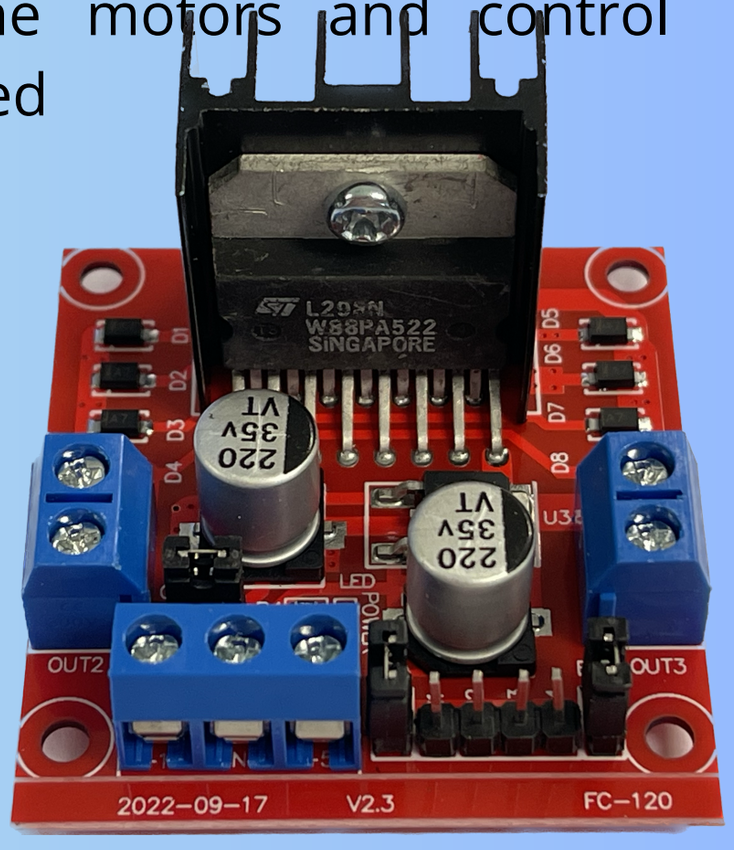
The L298N motor driver has a supply range of 5V to 35V and is capable of 2A continuous current per channel.

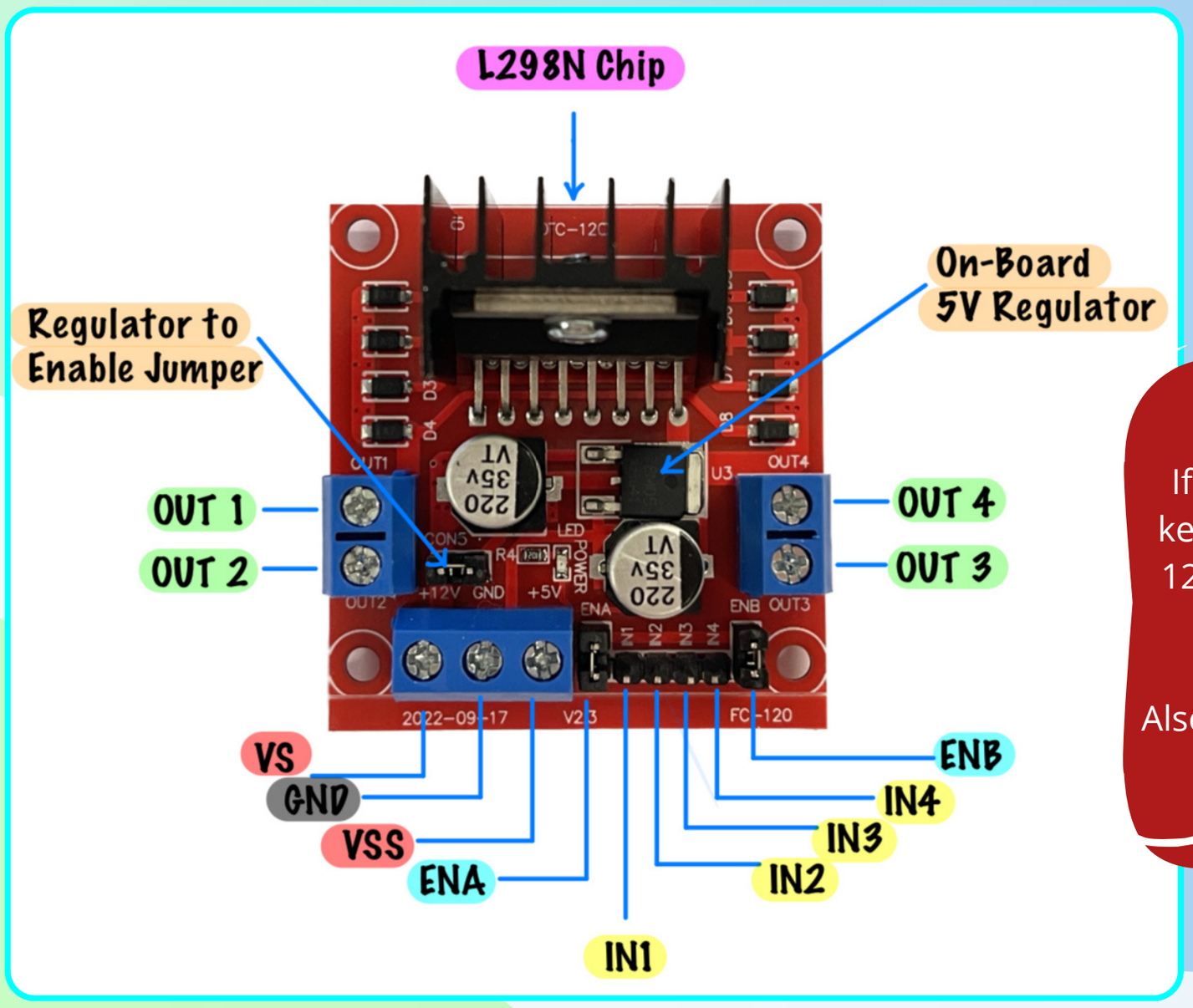
The module includes a 78M05 5V regulator. When a regulator jumper is in place, the 5V regulator is enabled, the logic power supply (VSS) is derived from the motor power supply (VS). The 5V input terminal acts as the output pin, delivering 5V 0.5A and can use it to power an Arduino or other circuitry.

When the jumper is removed, the 5V regulator is disabled, and we have to supply 5V separately through the VSS pin.

The L298N module has 11-pins that allow it to communicate with the outside world. The pinout is as follows :

- **VS** pin: Used to power the internal H-Bridge and drive the motor (5V-12V)
- **VSS** pin: Used to power the logic circuitry (5V-7V)
- **GND pin**: Ground (0V)
- **OUT1** and **OUT2** pin: For Motor A
- **OUT3** and **OUT4** pin: For Motor B
- **IN1** and **IN2** pin: Control the spinning direction of motor A
- **IN3** and **IN4** pin: Control the spinning direction of motor B
- **ENA** and **ENB** pin: Used to turn on/off the motors and control their speed





Warning:
If the motor power supply is less than 12V, keep the jumper in place. If it is greater than 12V, the jumper must be remove to prevent damage to the onboard 5V regulator.

Also, do not supply power to both the VSS and VS pins while the jumper is in place.

L298N Motor Driver Pinout

Geared DC Motor and Wheel

A **DC Motor** or direct current electrical motor is a rotating electro-mechanical device that turns electrical energy into mechanical energy.

Geared DC Motor is commonly used for making mobile robots. This motor can be powered with a power source as low as 2V and up to recommended voltage of 6V. The maximum voltage level of this DC motor is 12V.

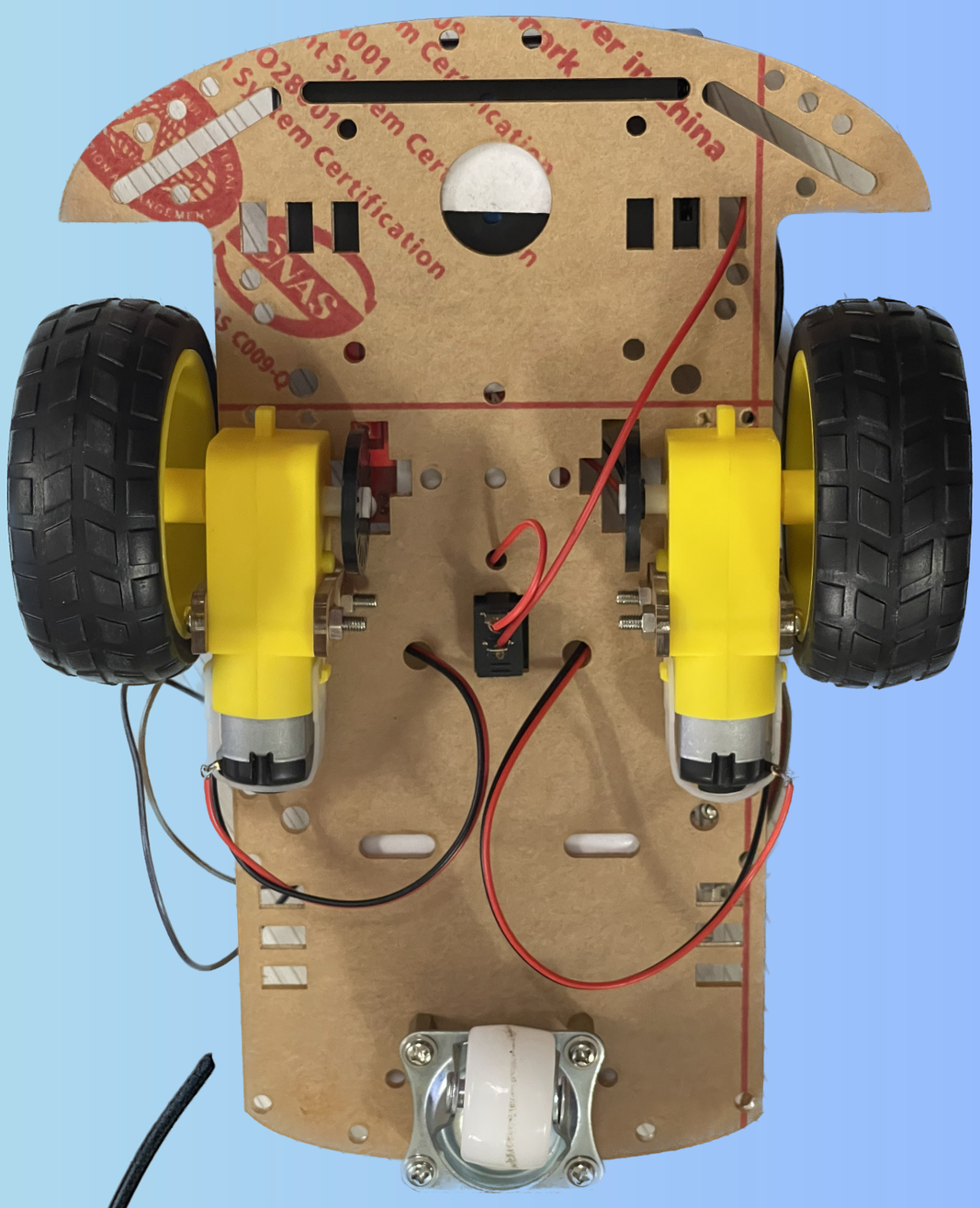
It has a dual output shaft, meaning one of the shafts can be attached to a wheel, gear or any attachment.

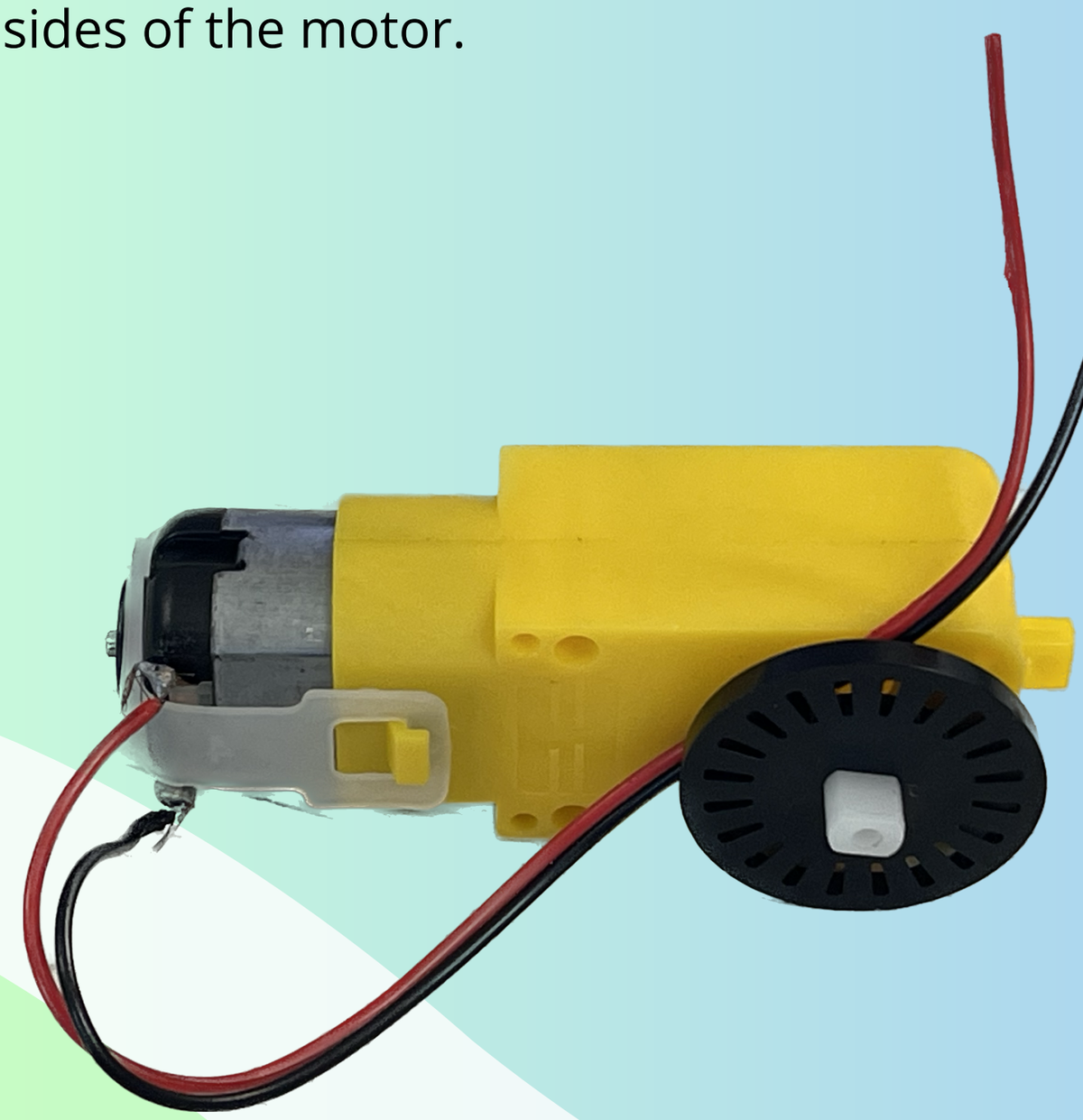
Another shaft attach to the rotary encoder wheel can be added a photointerrupter sensor to catch the speed of rotation.

The **wheel** diameter is 65 mm and the wheel can be attached at both sides of the motor.

The features of Geared DC Motor are such as below:

- Operating voltage: 3-12V
- Torque: 800 gf-cm min @ 3V
- Gear ratio: 1:48
- Min. operating speed (3V): 90+/- 10% RPM
- Min. operating speed (6V): 2000+/- 10% RPM





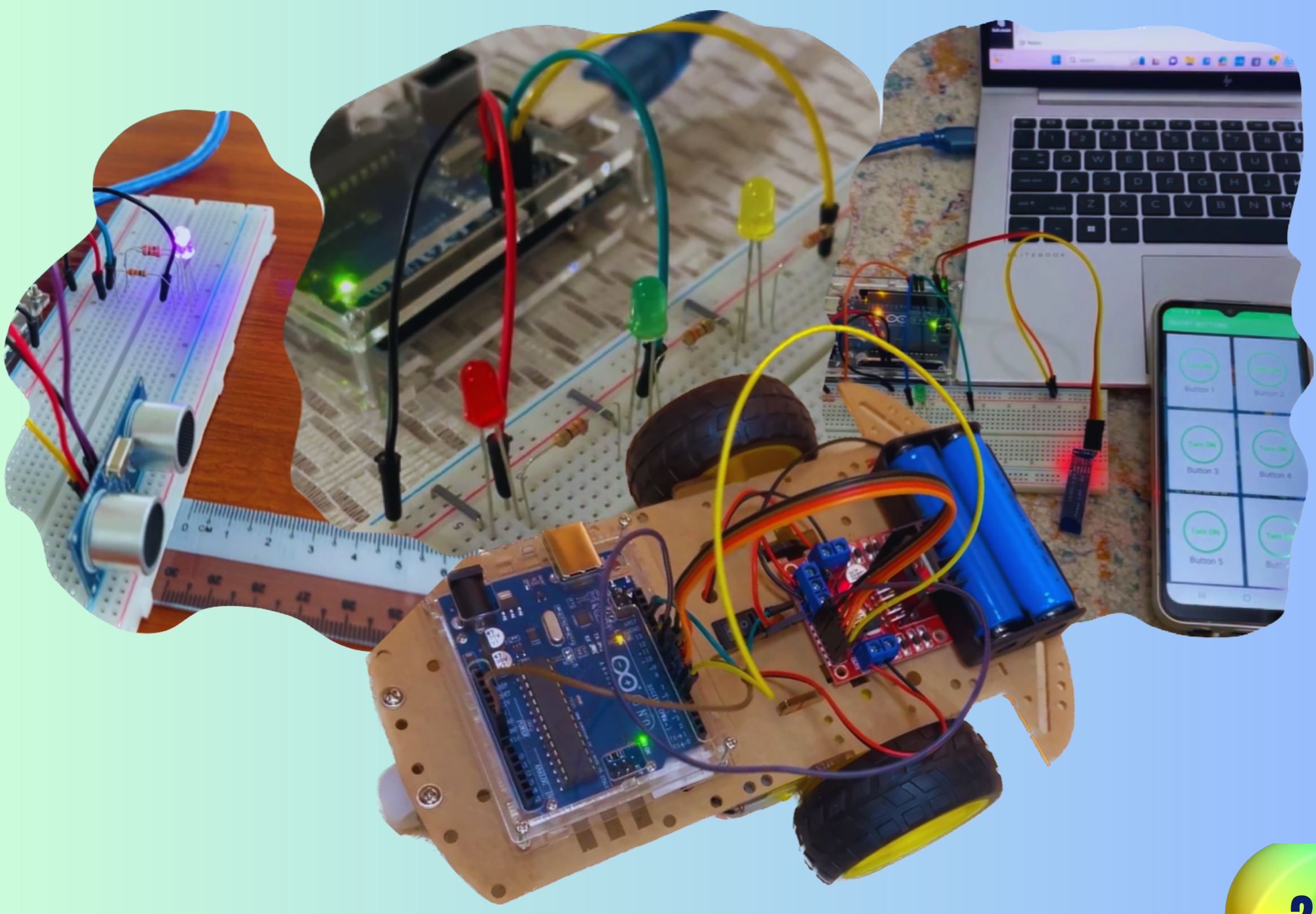
TOPIC

D

ARDUINO PROJECTS

There are ten (10) Arduino projects selected in these book as below:

1. **BLINKING LED**
2. **TRAFFIC LIGHT**
3. **CONTROL A RGB LED WITH BUTTON**
4. **AUTOMATIC STREET LIGHT**
5. **OBSTACLE DETECTION**
6. **DISTANCE DETECTOR**
7. **SOIL MOISTURE MONITORING**
8. **SMART HOME LIGHTING**
9. **AUTOMATIC CAR**
10. **BLUETOOTH CONTROLLED CAR**



PROJECT

1

BLINKING LED

Blinking LED project is as simple as turning a light on and off.

Understanding this project will give us a solid foundation as we work towards experiments that are more complex.

It uses code as an internal timer and continues to run until we cut the Arduino's power supply.

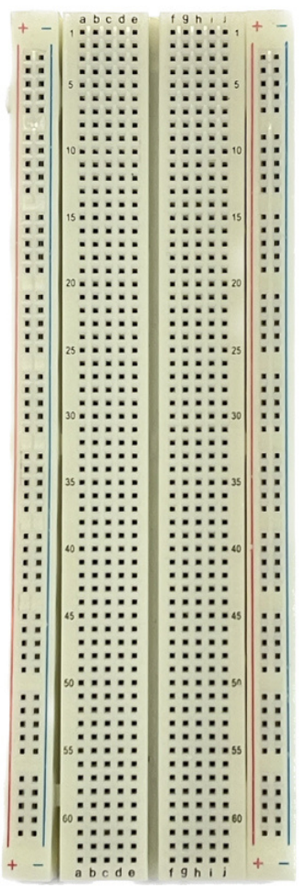
The LED will turn on for one second and turn off for another one second to make it blinking.

Component required for LED blinking project:

- 1 X Arduino Uno R3
- 1 X USB cable
- 1 X Breadboard
- 3 X Jumper wire
- 1 X LED
- 1 X 330 ohm Resistor

SCAN ME

USB CABLE



BREADBOARD

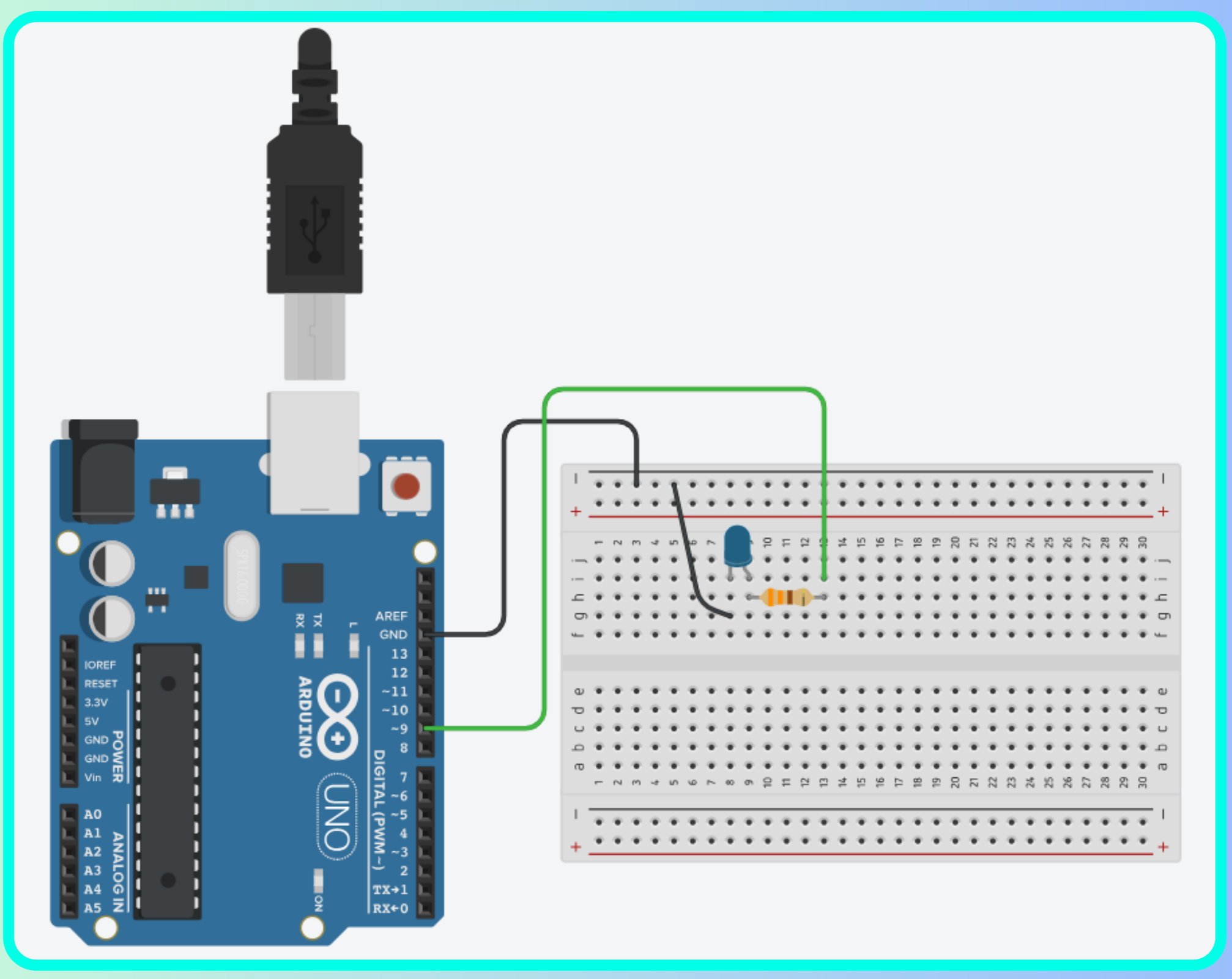
LED

RESISTOR (330 Ohm)

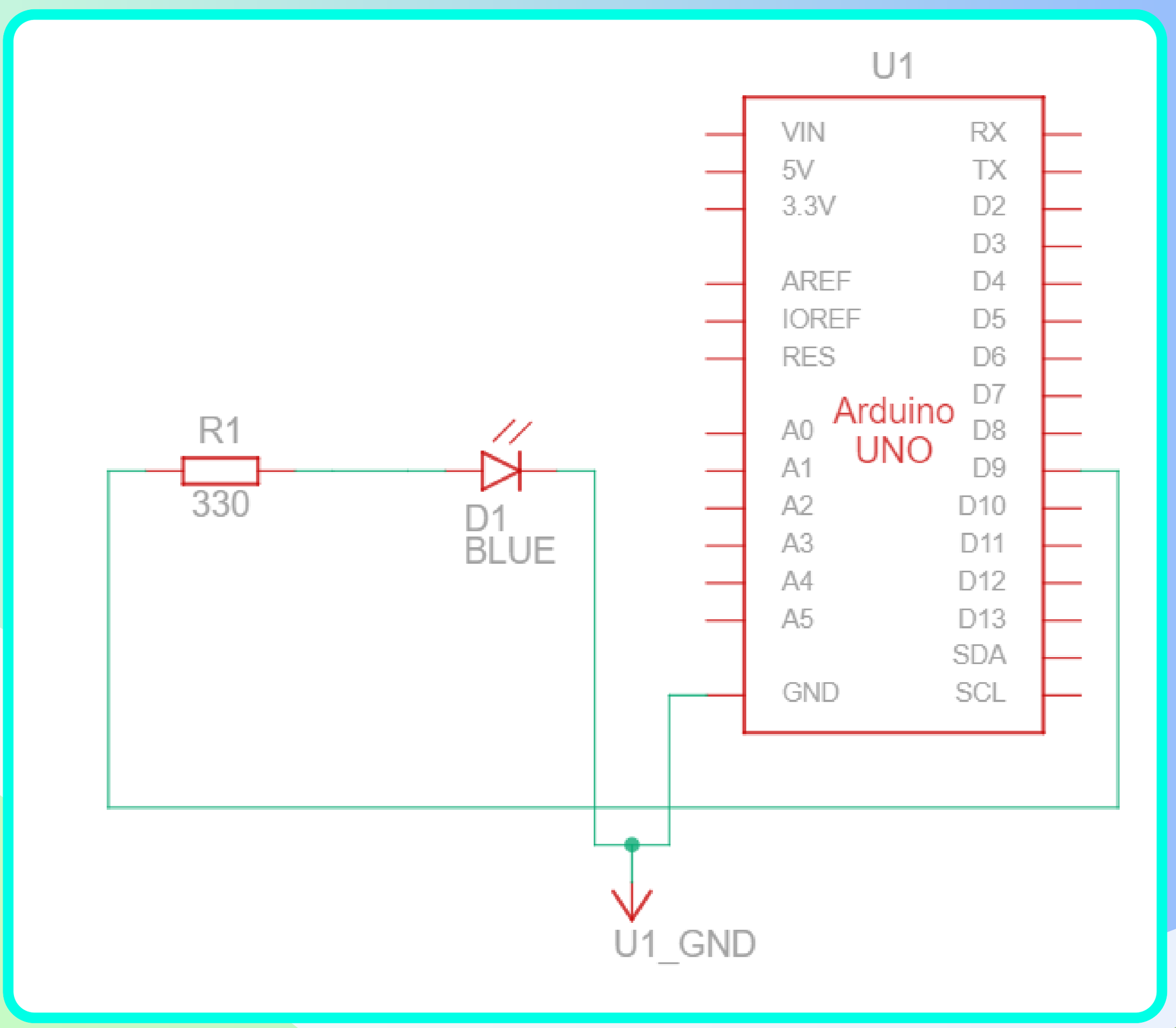
ARDUINO UNO

JUMPER WIRE (MALE TO MALE)

Components Required for Blinking LED



Tinkercad - Blinking LED Circuit Diagram



Schematic View - Blinking LED Circuit Diagram

```
1   // BLINKING LED
2
3   void setup() {
4
5     pinMode(9, OUTPUT);      // initialize digital pin 12 as an output.
6   }
7
8   // the loop function runs over and over again forever
9   void loop() {
10
11    digitalWrite(9, HIGH);   // turn the LED on (HIGH is the Voltage level)
12    delay(1000);              // wait for a 1000 milisecond
13    digitalWrite(9, LOW);    // turn the LED off by making the voltage LOW
14    delay(1000);
15
16  }
```

Arduino IDE Code for Blinking LED

PROJECT

2

TRAFFIC LIGHT

Traffic light project uses an Arduino Uno board and three LEDs to replicate a traffic light. A Green LED, red LED and yellow LED are used in this project.

It uses code as an internal timer and continues to run until we cut the Arduino's power supply.

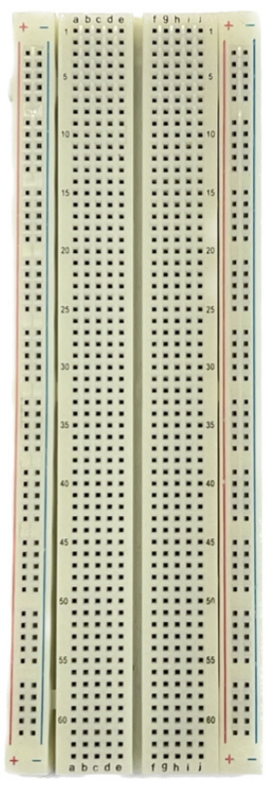
Set the red LED turn on for 5 seconds, followed by the green LED for 7 seconds and lastly, turn on the yellow LED for 3 seconds.

Component required for traffic light project:

- 1 X Arduino Uno R3
- 1 X USB cable
- 1 X Breadboard
- 7 X Jumper wire
- 1 X Red LED
- 1 X Green LED
- 1 X Yellow LED
- 3 X 330 ohm Resistor

SCAN ME

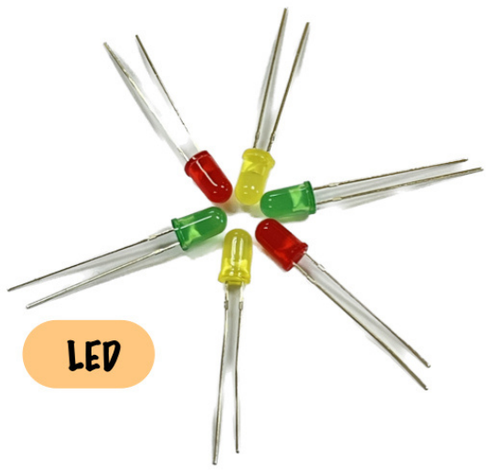
USB CABLE



BREADBOARD

JUMPER WIRE (MALE TO MALE)

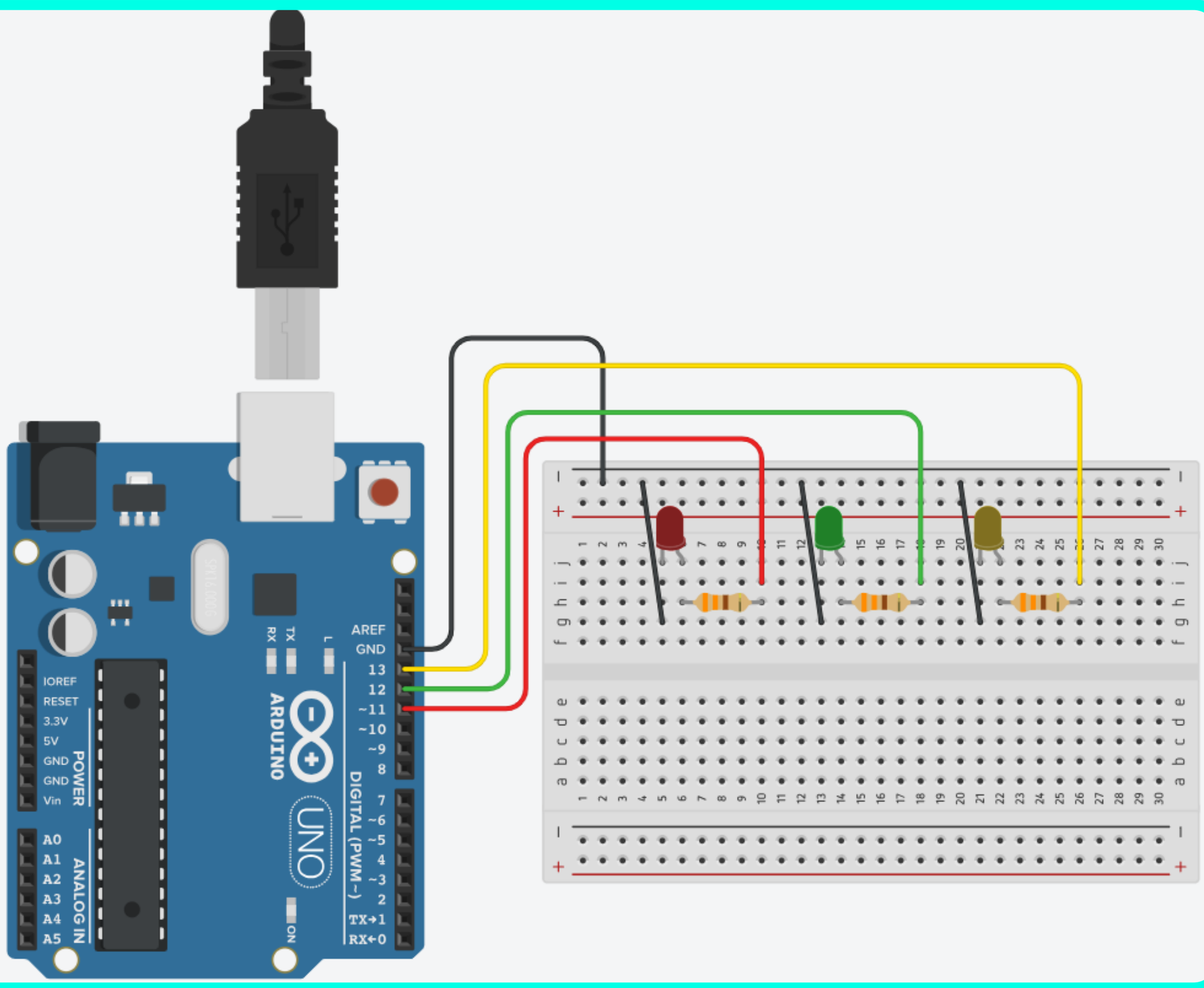
RESISTOR (330 Ohm)



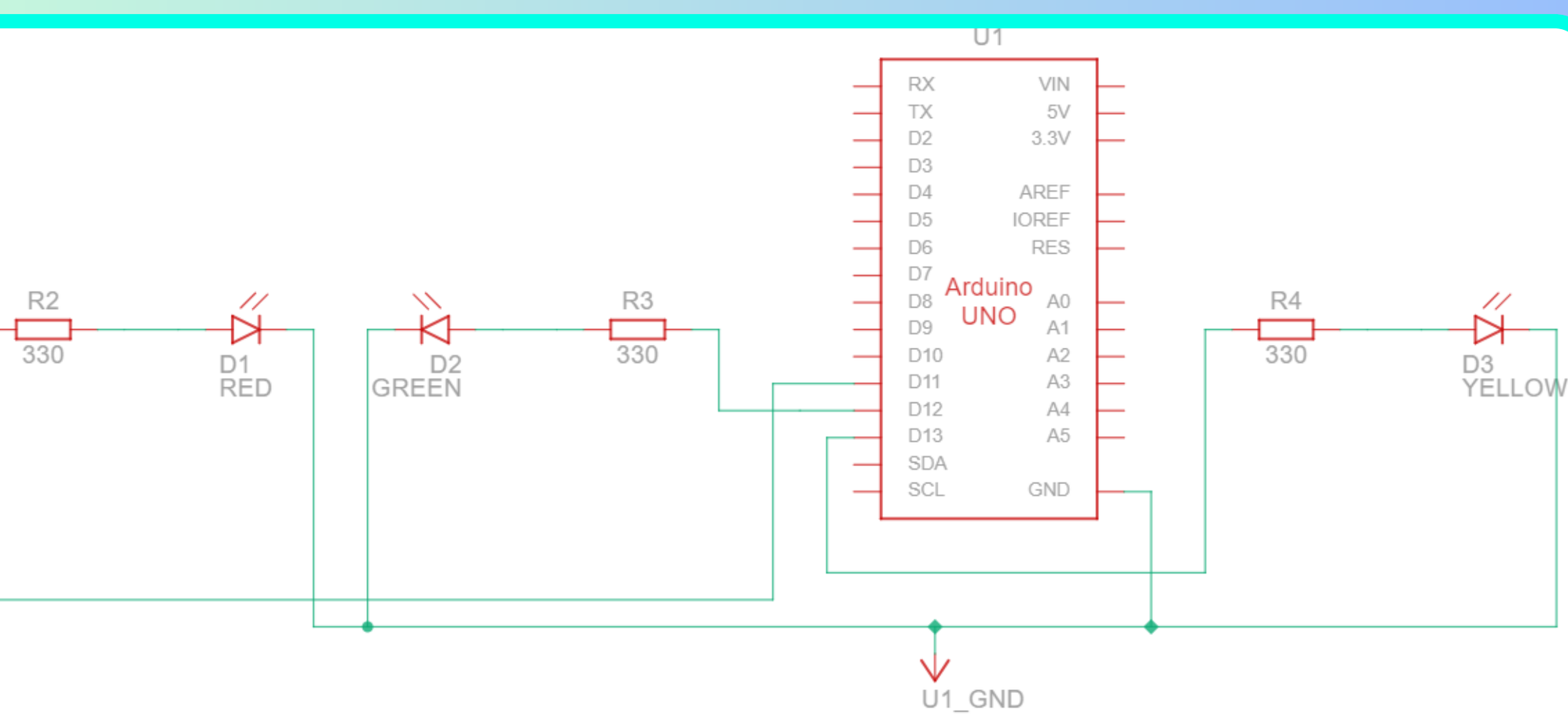
LED

ARDUINO UNO

Components Required for Traffic Light



Tinkercad - Traffic Light Circuit Diagram



Schematic View - Traffic Light Circuit Diagram

```ino
// TRAFFIC LIGHT

void setup() {

  pinMode(11, OUTPUT);  // Initialize digital pin 11 (Red LED) as an output.
  pinMode(12, OUTPUT);  // Initialize digital pin 12 (Green LED) as an output.
  pinMode(13, OUTPUT);  // Initialize digital pin 13 (Yellow LED) as an output.
}

// the loop function runs over and over again forever.
void loop() {
  digitalWrite(11, HIGH);  // Turn the Red LED on (HIGH is the Voltage level)
  delay(5000);              // Wait for a 5 second
  digitalWrite(11, LOW);  // Turn the LED off by making the voltage LOW
  delay(500);

  digitalWrite(12, HIGH);  // Turn the Green LED on (HIGH is the Voltage level)
  delay(7000);              // Wait for a 7 second
  digitalWrite(12, LOW);  // Turn the LED off by making the voltage LOW
  delay(500);

  digitalWrite(13, HIGH);  // Turn the Yellow LED on (HIGH is the Voltage level)
  delay(3000);              // Wait for a 3 second
  digitalWrite(13, LOW);  // Turn the LED off by making the voltage LOW
  delay(500);
}
```

Arduino IDE Code for Traffic Light

PROJECT

3

CONTROL LED WITH BUTTON

Control LED with button project uses an Arduino Uno board, a push button and one RGB LED.

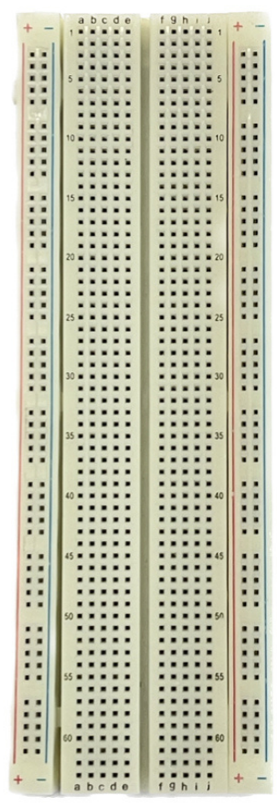
In this project, we will control RGB LED using push button to change color interfacing with Arduino Uno board.

The RGB LED will cycle through the colors white, pink, dark green, navy blue, red and purple when we press the push button.

Component required for controlling a RGB LED with Button project:

- 1 X Arduino Uno R3
- 1 X USB cable
- 1 X Breadboard
- 6 X Jumper wire
- 1 X LED RGB
- 3 X 330 ohm Resistor
- 1 X 1K ohm Resistor
- 1 X Piezo-Buzzer Module

SCAN ME

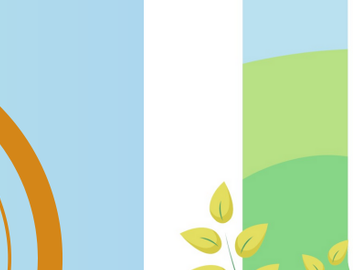


BREADBOARD

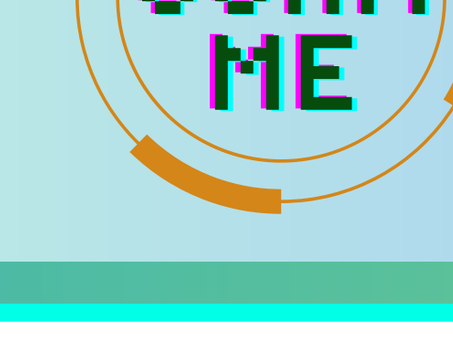
RBG LED

RESISTOR (330 Ohm)

RESISTOR (1K Ohm)



PUSH BUTTON

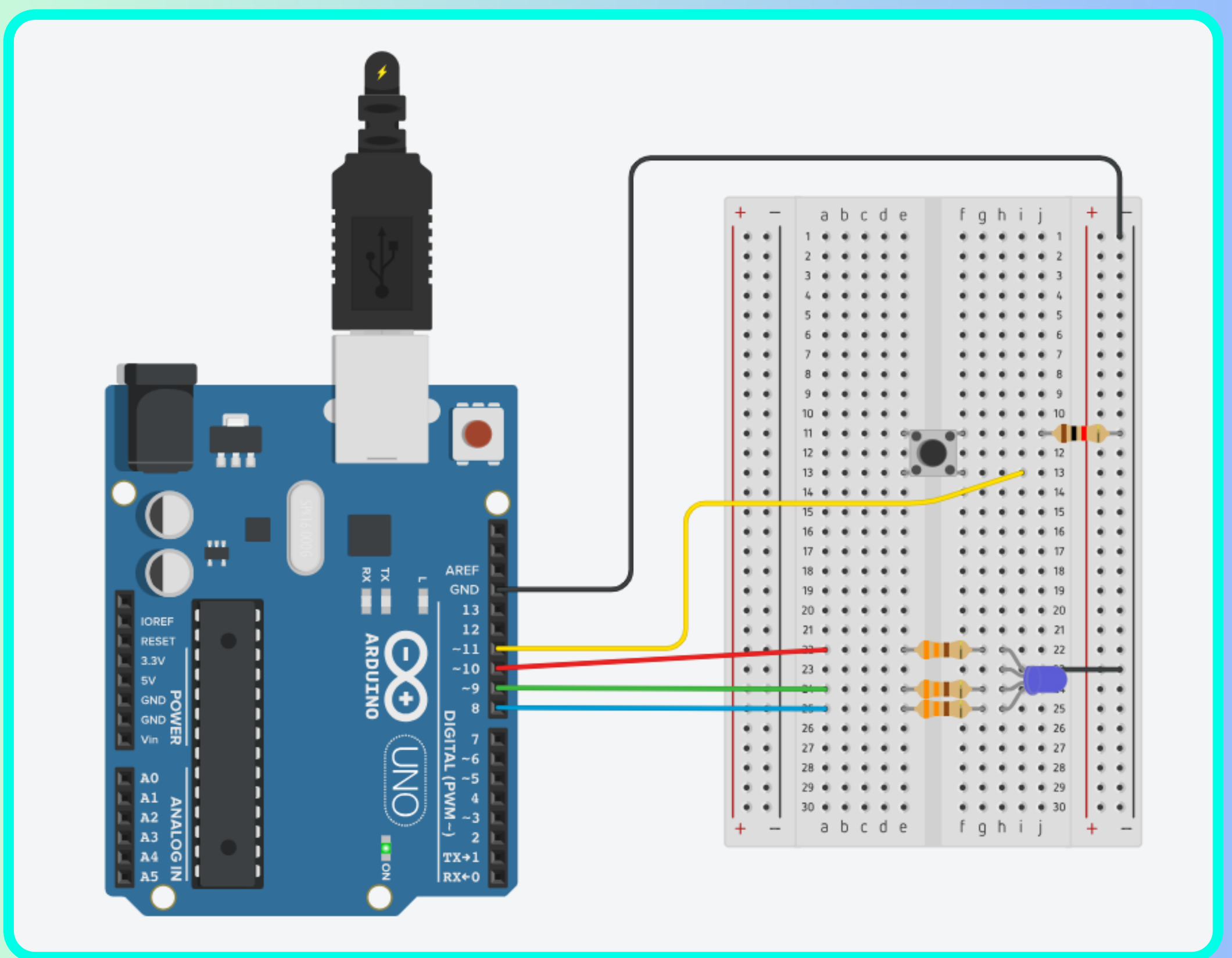


JUMPER WIRE (MALE TO MALE)

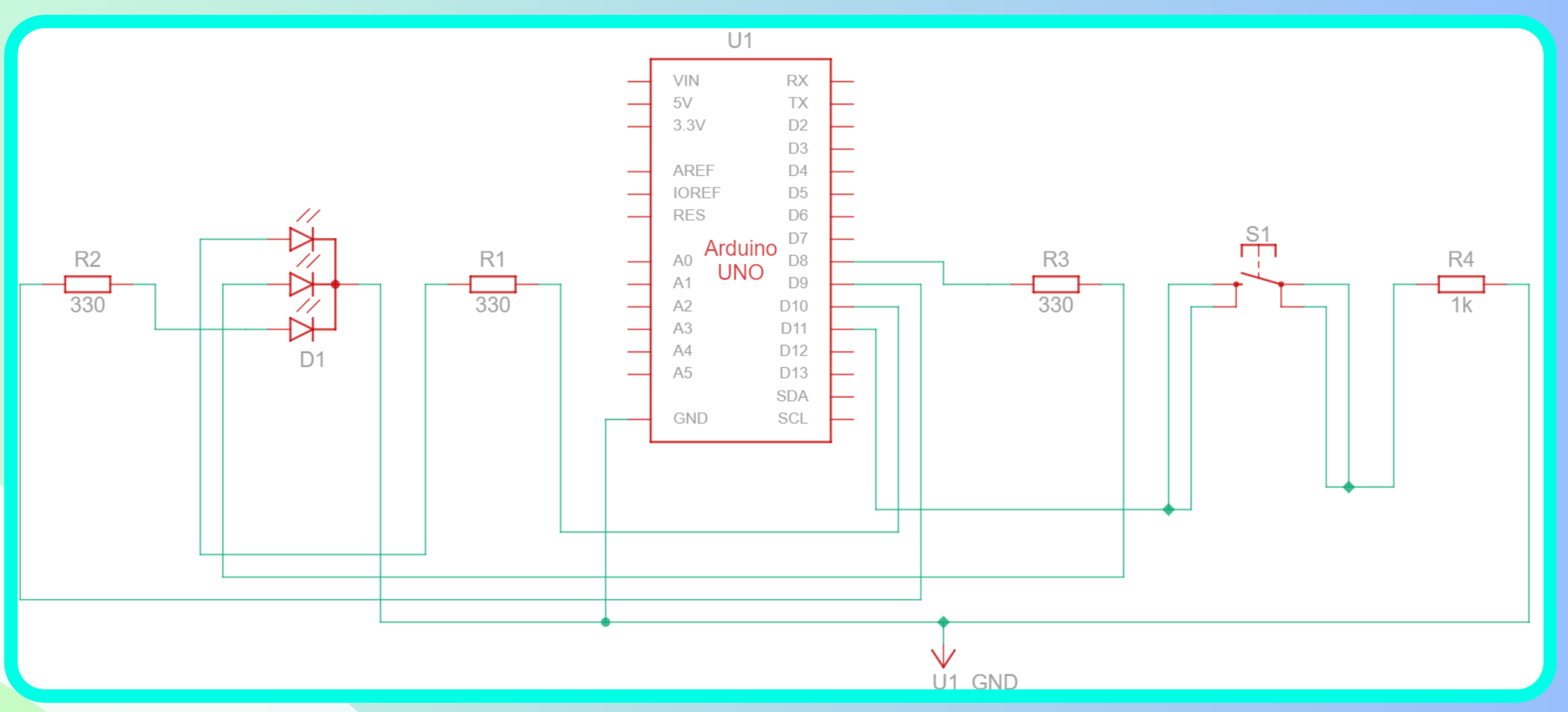
USB CABLE

ARDUINO UNO

Components Required for Control LED with Button



Tinkercad - Control LED with Button Circuit Diagram



Schematic View - Control LED with Button Circuit Diagram

```
// CONTROL A RGB LED WITH BUTTON

// Define pins
#define red 10
#define green 9
#define blue 8
#define button 11

//Define color mode
int mode = 0;

// Setup LED Light dan Buttons

void setup() {
  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
  pinMode(blue, OUTPUT);
  pinMode(button, INPUT_PULLUP);
}

void loop(){
  if(digitalRead(button) == LOW){
    mode = mode + 1;
    delay(400);
  }

// OFF
  if (mode == 0){
    analogWrite(red, 0);
    analogWrite(green, 0);
    analogWrite(blue, 0);
  }

// WHITE
  if (mode == 1){
    analogWrite(red, 255);
    analogWrite(green, 255);
    analogWrite(blue, 255);
  }

// PINK
  if (mode == 2){
    analogWrite(red, 255);
    analogWrite(green, 20);
    analogWrite(blue, 147);

  }

```

Arduino IDE Code for Control a RGB LED with Button

```
// DARK GREEN
  if (mode == 3){
    analogWrite(red, 0);
    analogWrite(green, 100);
    analogWrite(blue, 0);
  }

// NAVY BLUE
  if (mode == 4){
    analogWrite(red, 0);
    analogWrite(green, 0);
    analogWrite(blue, 128);
  }

// RED
  if (mode == 5){
    analogWrite(red, 225);
    analogWrite(green, 0);
    analogWrite(blue, 0);
  }

// PURPLE
  if (mode == 6){
    analogWrite(red, 128);
    analogWrite(green, 0);
    analogWrite(blue, 128);
  }

// SWITCH OFF
  if (mode == 7){
    mode = 0;
  }
}
```

Arduino IDE Code for Control a RGB LED with Button

PROJECT

4

AUTOMATIC STREET LIGHT

Automatic street light works depend on the presence or absence of light in the atmosphere. LDR was chosen in this project to read the intensity of light and it works extremely well in managing light.

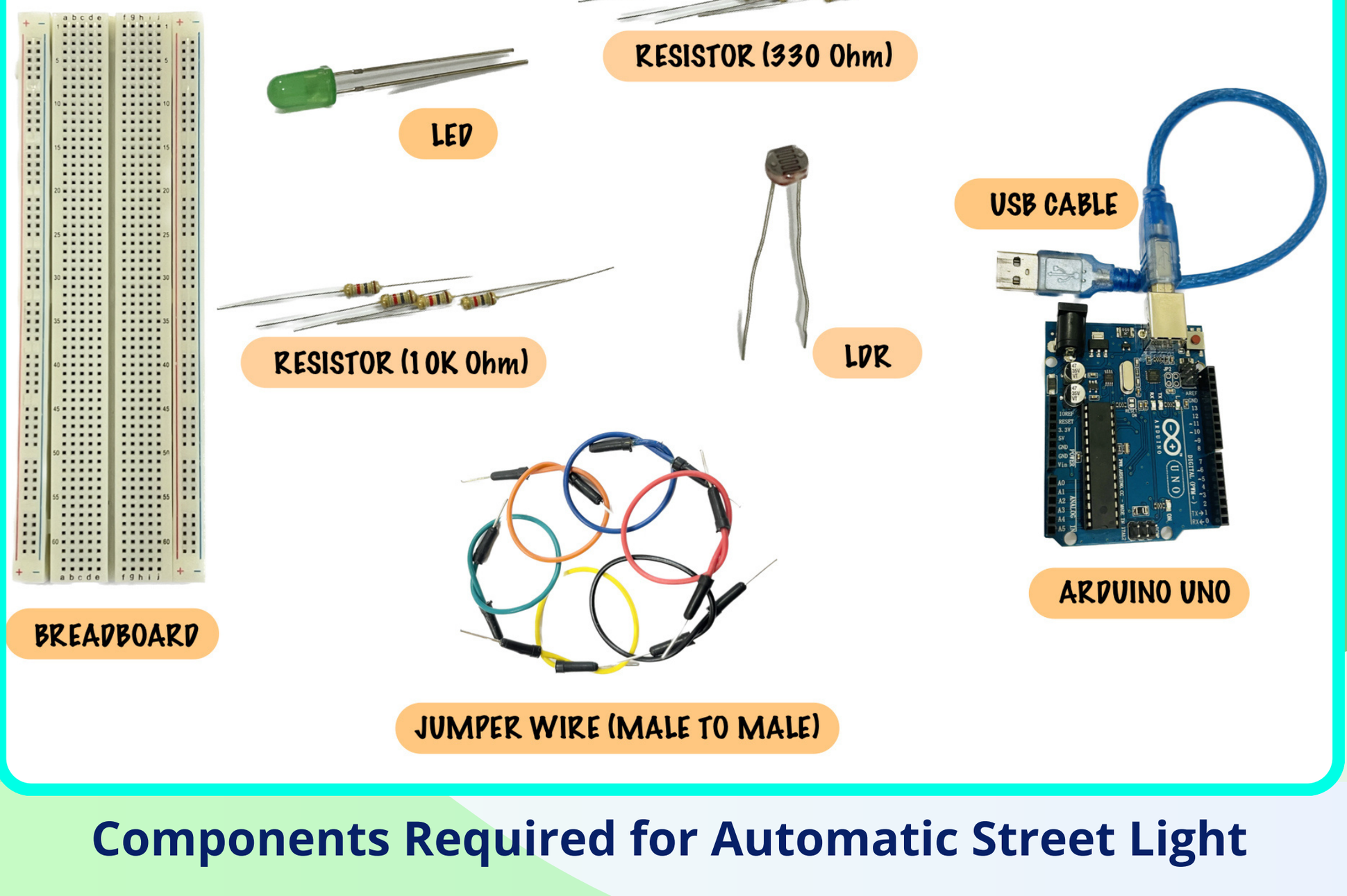
When darkness rises to a certain level, then the sensor circuit gets activated and switches on the LED. When there is another source of light, for example during the daytime, the LED gets off.

LDR is usually reasonably priced and this is a clever way of bringing automation into the street light system.

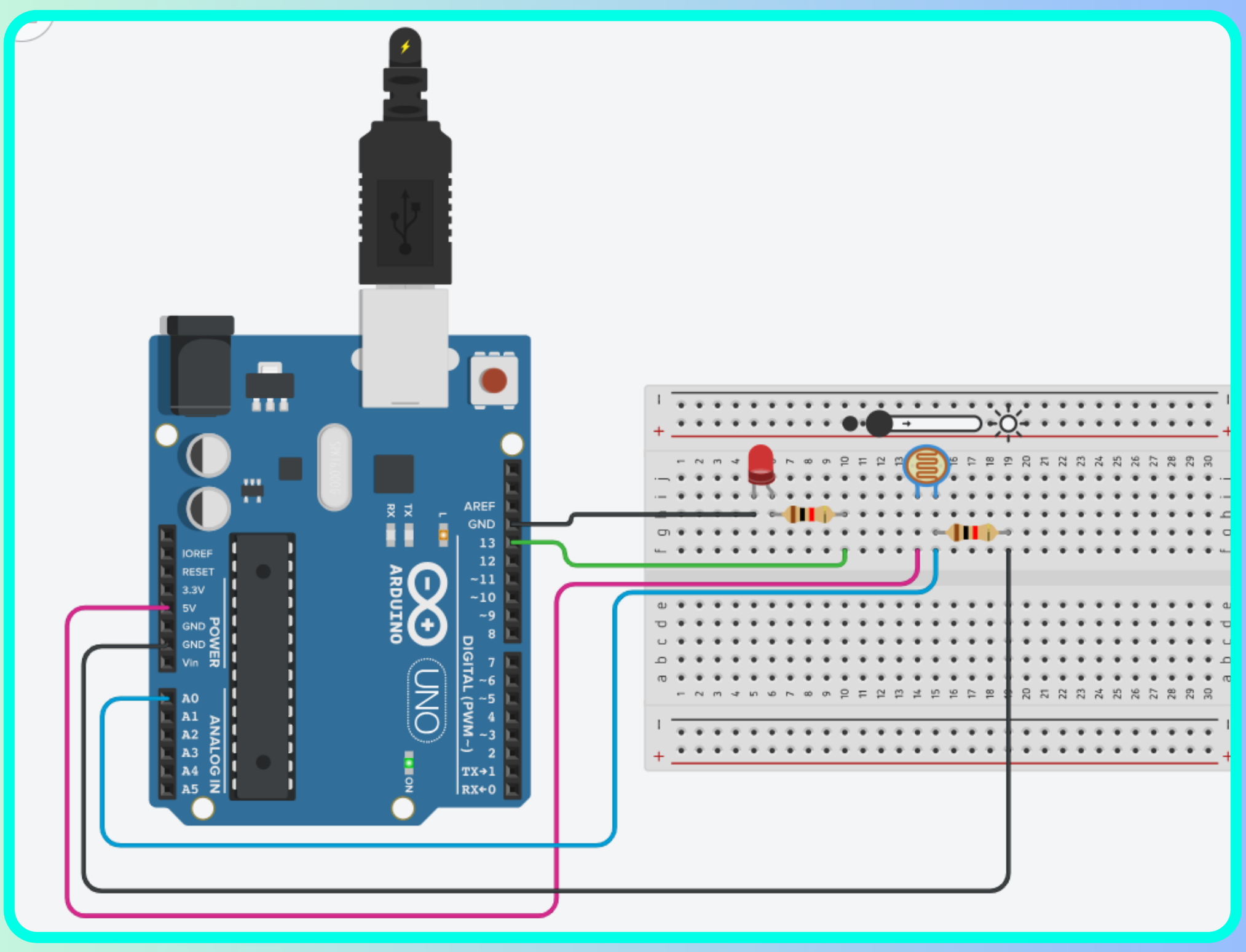
Component required for Automatic Street Light:

- 1 X Arduino Uno R3
- 1 X USB cable
- 1 X Breadboard
- 5 X Jumper wire
- 1 X LED
- 1 X LDR
- 1 X 10K ohm Resistor
- 1 X 330 ohm Resistor

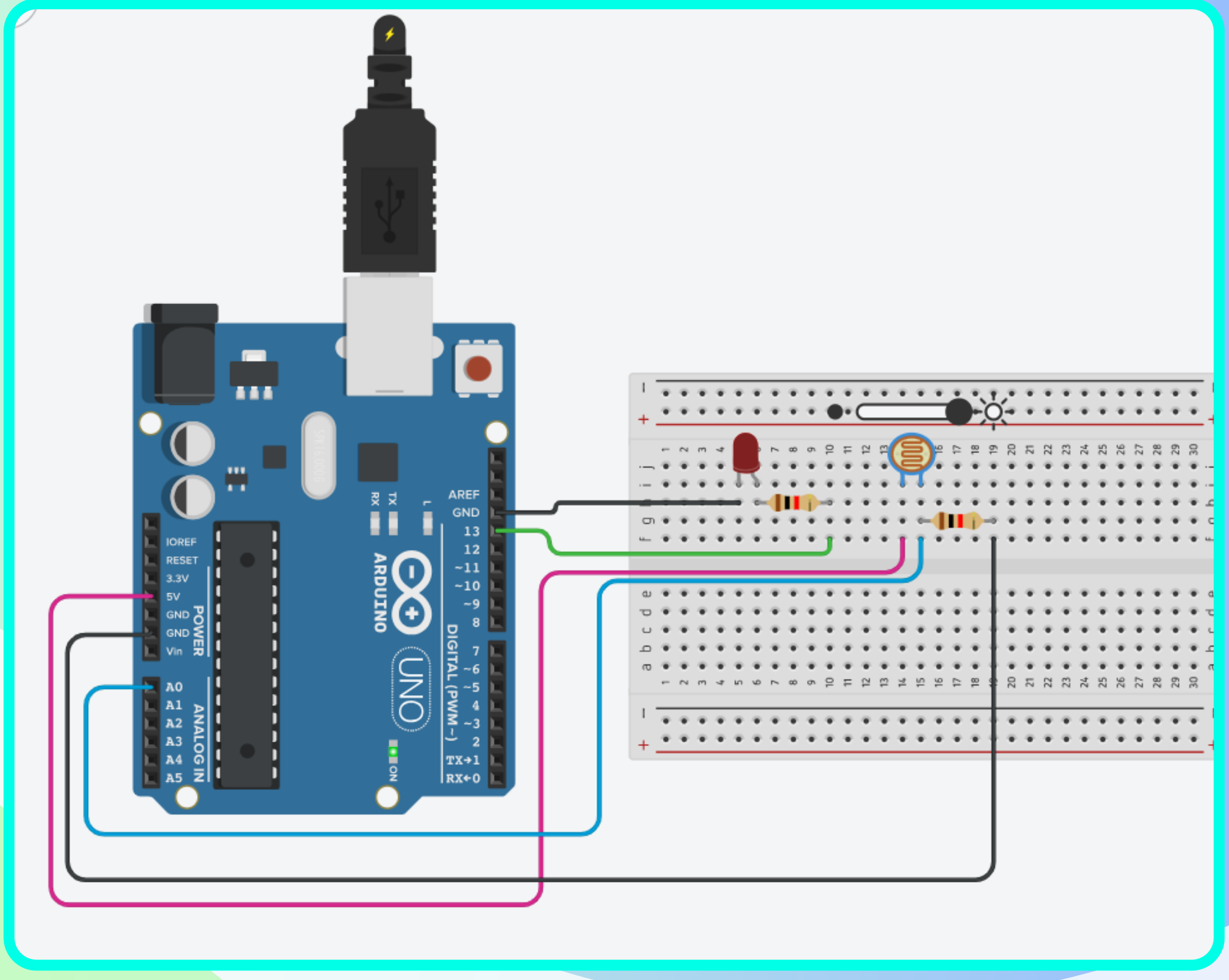
SCAN ME



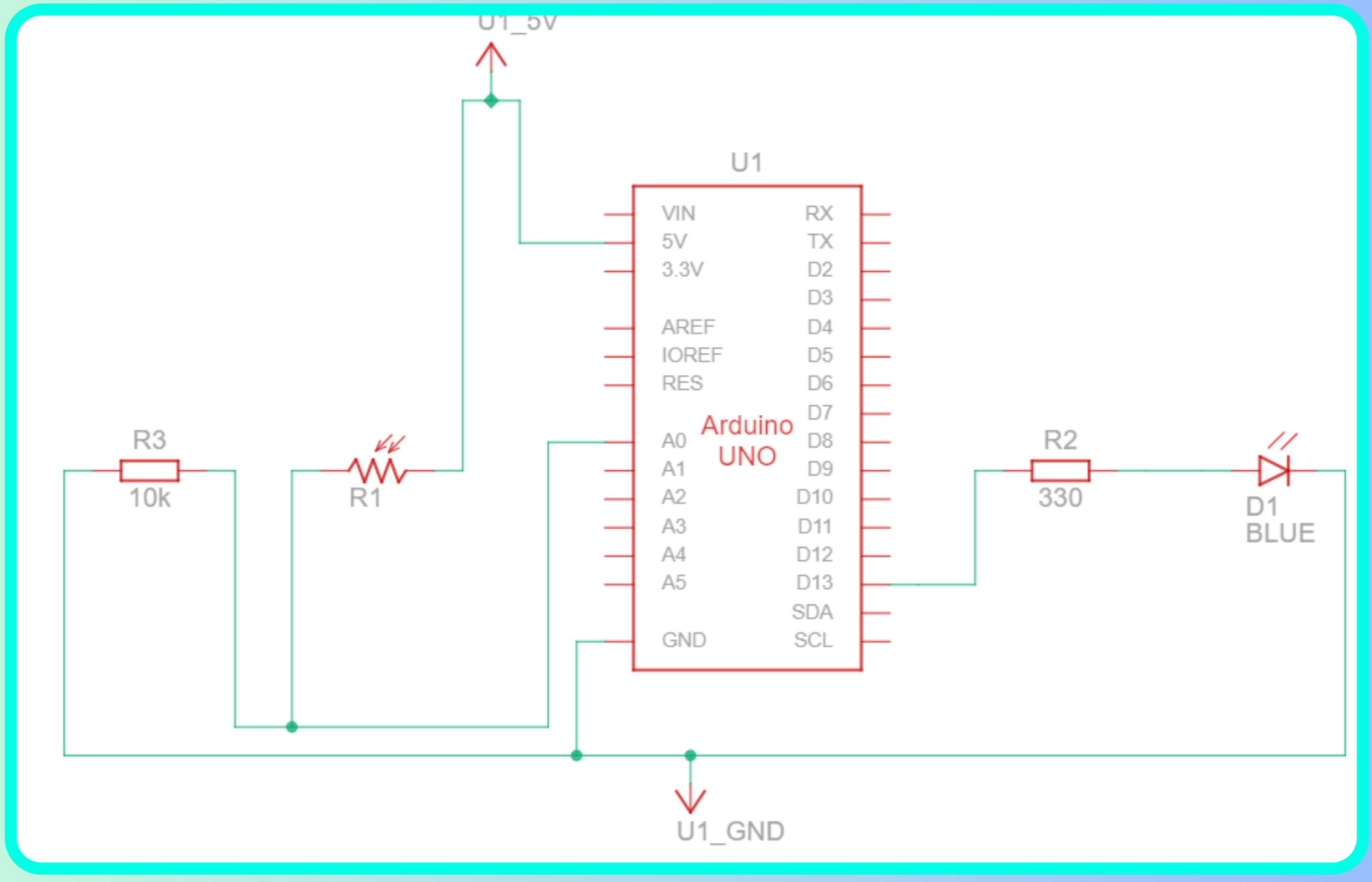
Components Required for Automatic Street Light



Tinkercad - Automatic Street Light Circuit Diagram when LED is on



Tinkercad - Automatic Street Light Circuit Diagram when LED is off



Schematic View - Automatic Street Light Circuit Diagram

```
// AUTOMATIC STREET LIGHT

const int ledPin = 13;   //The number of the LED pin
const int ldrPin = A0;  //The number of the LDR pin

void setup() {

  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);  //Initialize the LED pin as an output
  pinMode(ldrPin, INPUT);   //Initialize the LDR pin as an input
}

void loop() {

  int ldrStatus = analogRead(ldrPin);   //Read the status of the LDR value

  //check if the LDR status is <= 300
  //if it is, the LED is HIGH

   if (ldrStatus <=300) {

    digitalWrite(ledPin, HIGH);                //Turn LED on
    Serial.println("LDR is DARK, LED is ON");

   }
  else {

    digitalWrite(ledPin, LOW);                 //Turn LED off
    Serial.println("LED is OFF");
  }
}
```

Arduino IDE Code for Automatic Street Light

PROJECT

5

OBSTACLE DETECTION

Obstacle Detection is applicable to anything that moves, including robot manipulators and vehicles.

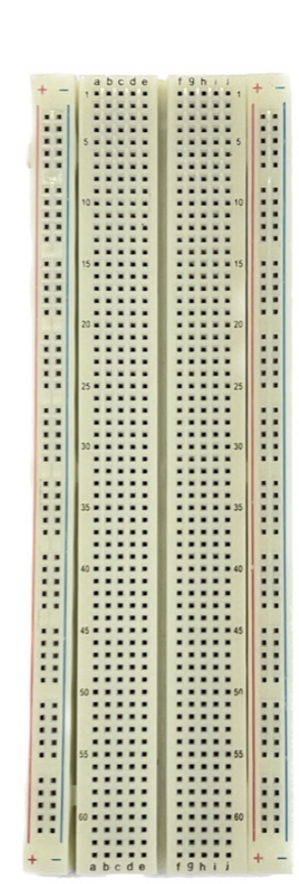
Obstacle detection can sense slow-moving or stationary objects when driving at low speeds.

This project uses infrared (IR) sensors to detect the obstacle. The LED will light up and the buzzer will be on when there is an obstacle in front of the sensor.

Component required for obstacle detection project:

- 1 X Arduino Uno R3
- 1 X USB cable
- 1 X Breadboard
- 8 X Jumper wire
- 1 X LED
- 1 X Buzzer
- 1 X Infrared (IR) sensor
- 1 X 330 ohm Resistor

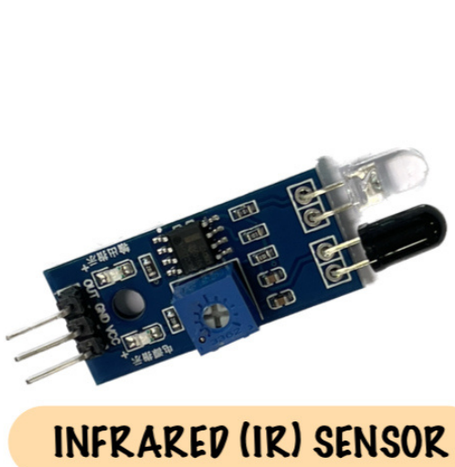
SCAN ME



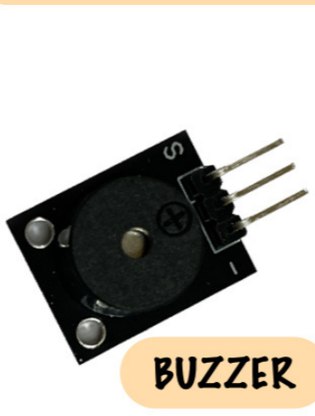
BREADBOARD

LED

RESISTOR (330 Ohm)



INFRARED (IR) SENSOR



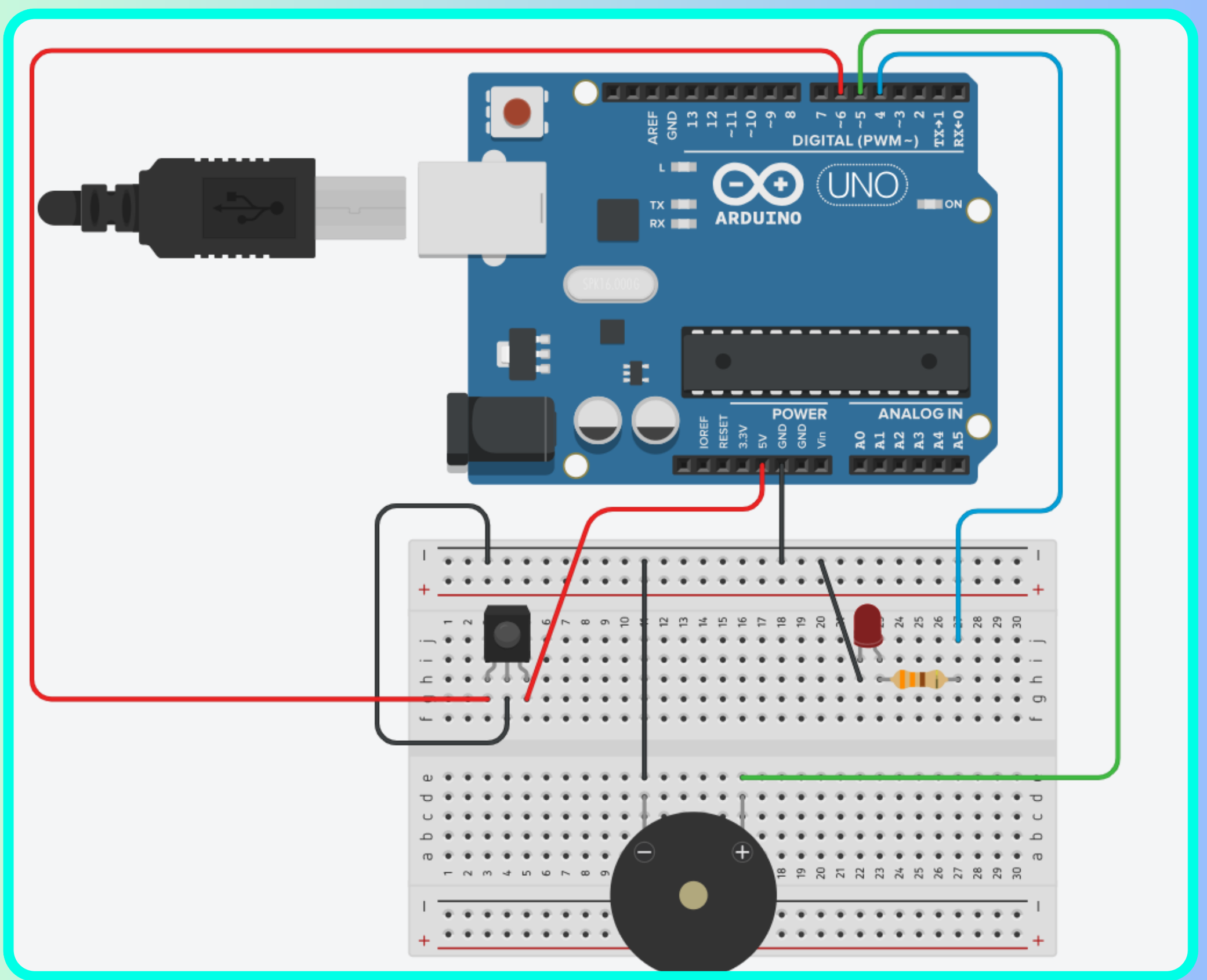
BUZZER

JUMPER WIRE (MALE TO MALE)

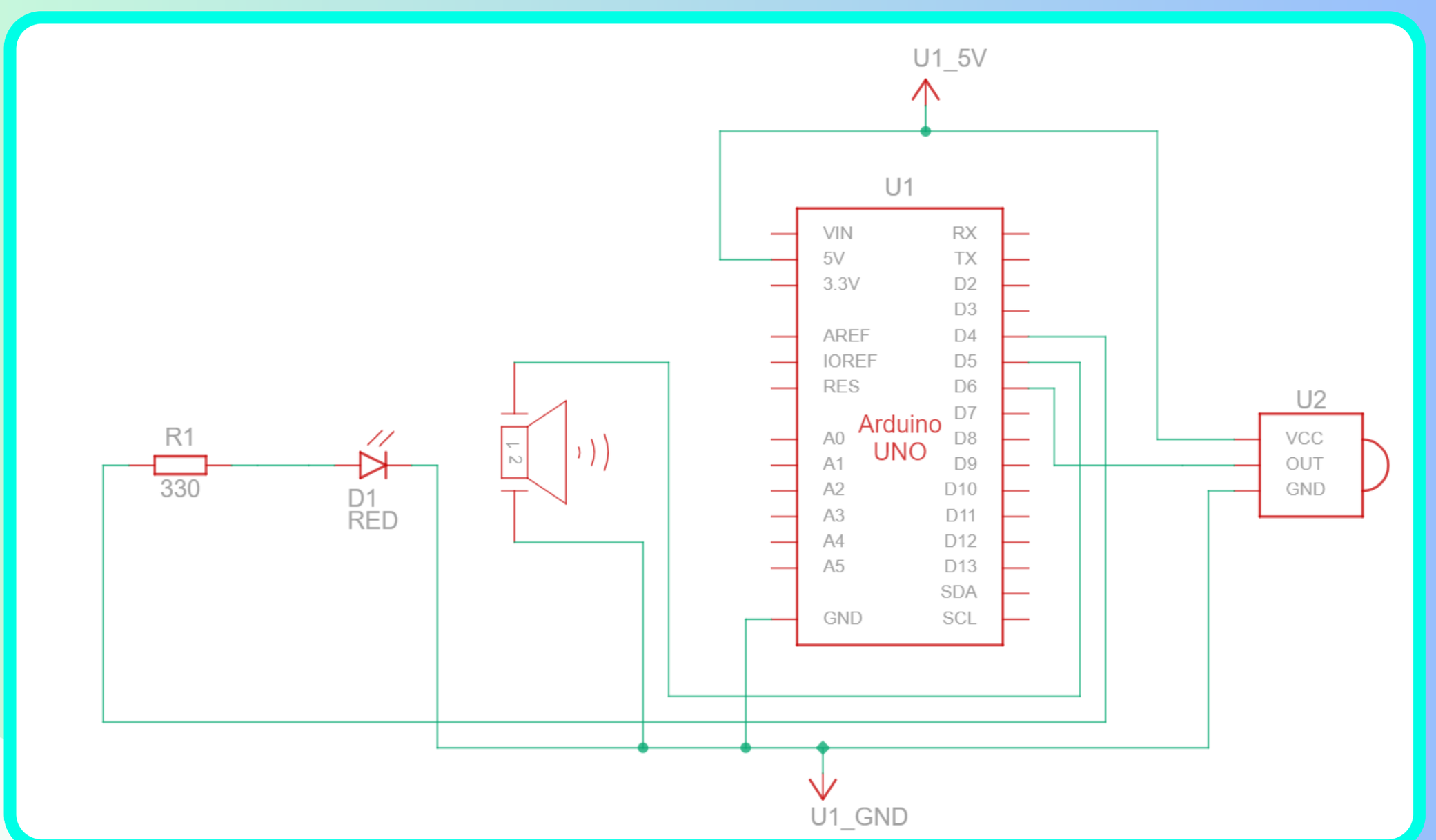
USB CABLE

ARDUINO UNO

Components Required for Obstacle Detection



Tinkercad - Obstacle Detection Circuit Diagram



Schemativ View - Obstacle Detection Circuit Diagram

```
// OBSTACLE DETECTION

int irPin = 6;
int buzzerPin = 5;
int ledPin = 4;
int irVal;

void setup(){
  pinMode(irPin, INPUT);
  pinMode(buzzerPin, OUTPUT);
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}

void loop(){
  irVal = digitalRead(irPin);

  if(irVal == 0){
    Serial.println("Obstacle Detected");
    digitalWrite(buzzerPin, HIGH);
    digitalWrite(ledPin, HIGH);
  }

  else{
    Serial.println("No Obstacle Detected");
    digitalWrite(buzzerPin, LOW);
    digitalWrite(ledPin, LOW);
  }
}
```

Arduino IDE Code for Obstacle Detection

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on 'COM16')

```
22:45:23.499 -> No Obstacle Detected
22:45:23.701 -> No Obstacle Detected
22:45:23.865 -> No Obstacle Detected
22:45:24.066 -> No Obstacle Detected
22:45:24.233 -> No Obstacle Detected
22:45:24.434 -> No Obstacle Detected
22:45:24.610 -> No Obstacle Detected
22:45:24.811 -> No Obstacle Detected
22:45:24.977 -> No Obstacle Detected
```

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on 'COM16')

```
22:46:04.663 -> Obstacle Detected
22:46:04.832 -> Obstacle Detected
22:46:04.963 -> Obstacle Detected
22:46:05.131 -> Obstacle Detected
22:46:05.300 -> Obstacle Detected
22:46:05.442 -> Obstacle Detected
22:46:05.606 -> Obstacle Detected
22:46:05.774 -> Obstacle Detected
22:46:05.941 -> Obstacle Detected
```

Serial Monitor

PROJECT

6

DISTANCE DETECTOR

Distance Detector uses an Ultrasonic Sensor for determining the distance of an object from another object without any physical contact involved.

RGB LED will be used as an output in this project. LED will change its color depending on the distance of the object.

When the object is placed at a distance of more than 16cm, the RGB LED will light up in white color.

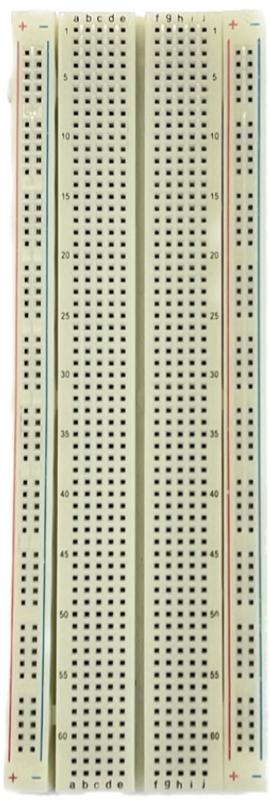
If the distance is less than 15cm, the RGB LED will light up in an orange color, distance less than 10cm, the RGB LED will light up a magenta color and lastly when the distance is less than 5cm, the RGB LED will light up blue color.

LED and buzzer will ON when the soil is dry dan OFF when the soil has sufficient moisture (wet)

Component required for distance detector project:

- 1 X Arduino Uno R3
- 1 X USB cable
- 1 X Breadboard
- 9 X Jumper wire
- 1 X RGB LED
- 1 X Ultrasonic Sensor
- 3 X 330 ohm Resistor

SCAN ME



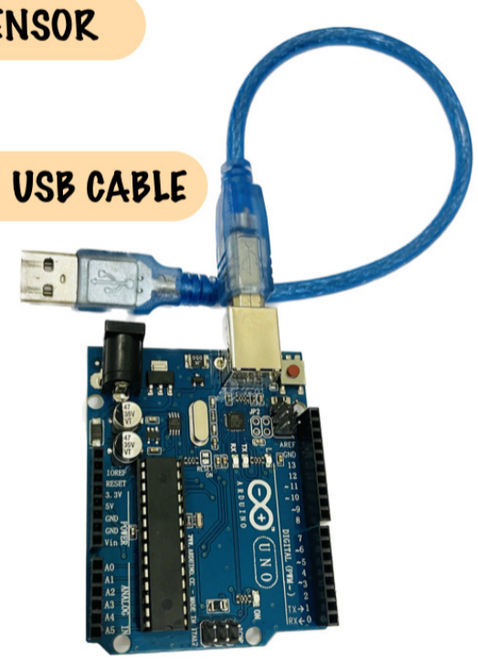
BREADBOARD

RESISTOR (330 Ohm)

ULTRASONIC SENSOR

RGB LED

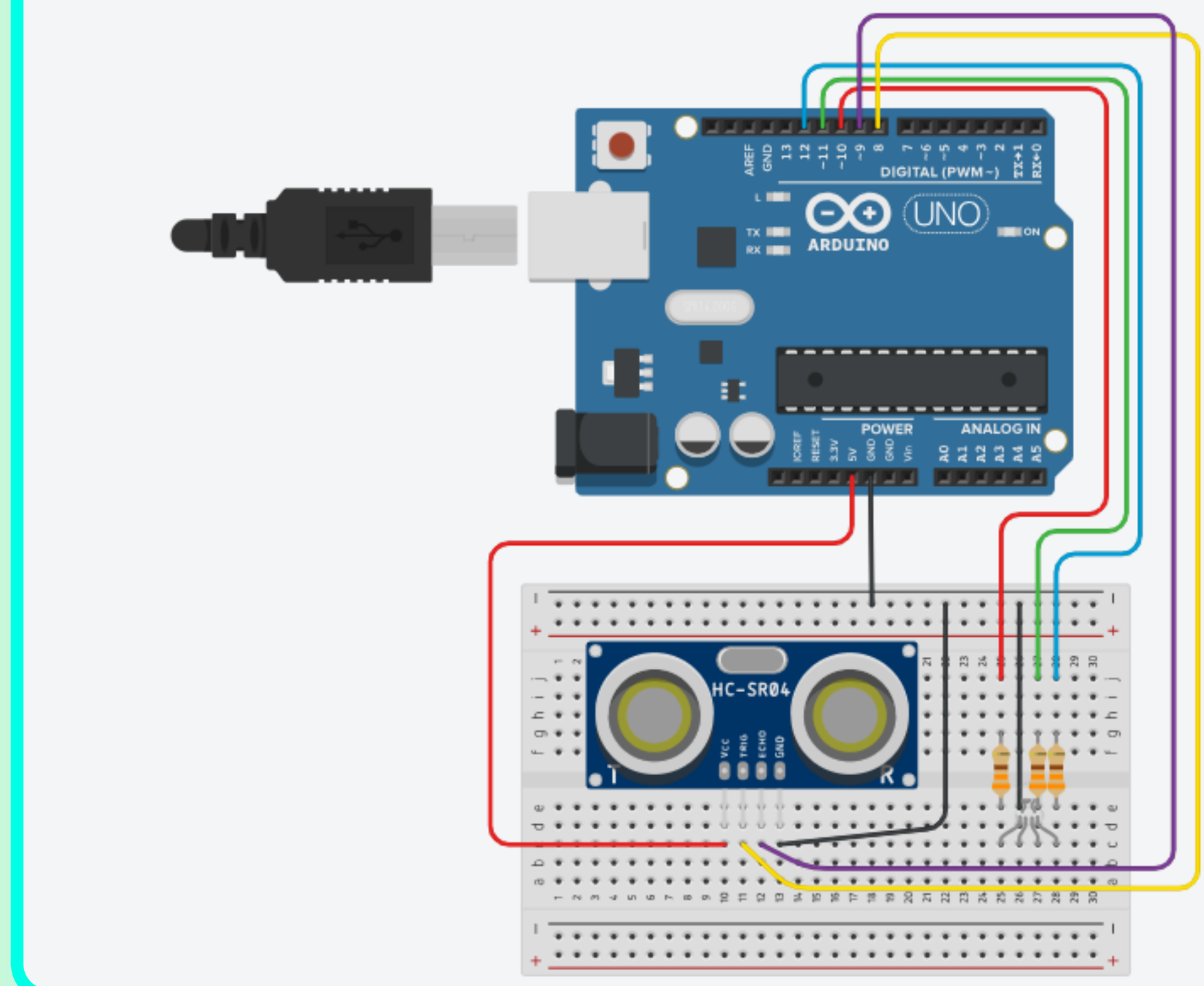
JUMPER WIRE (MALE TO MALE)



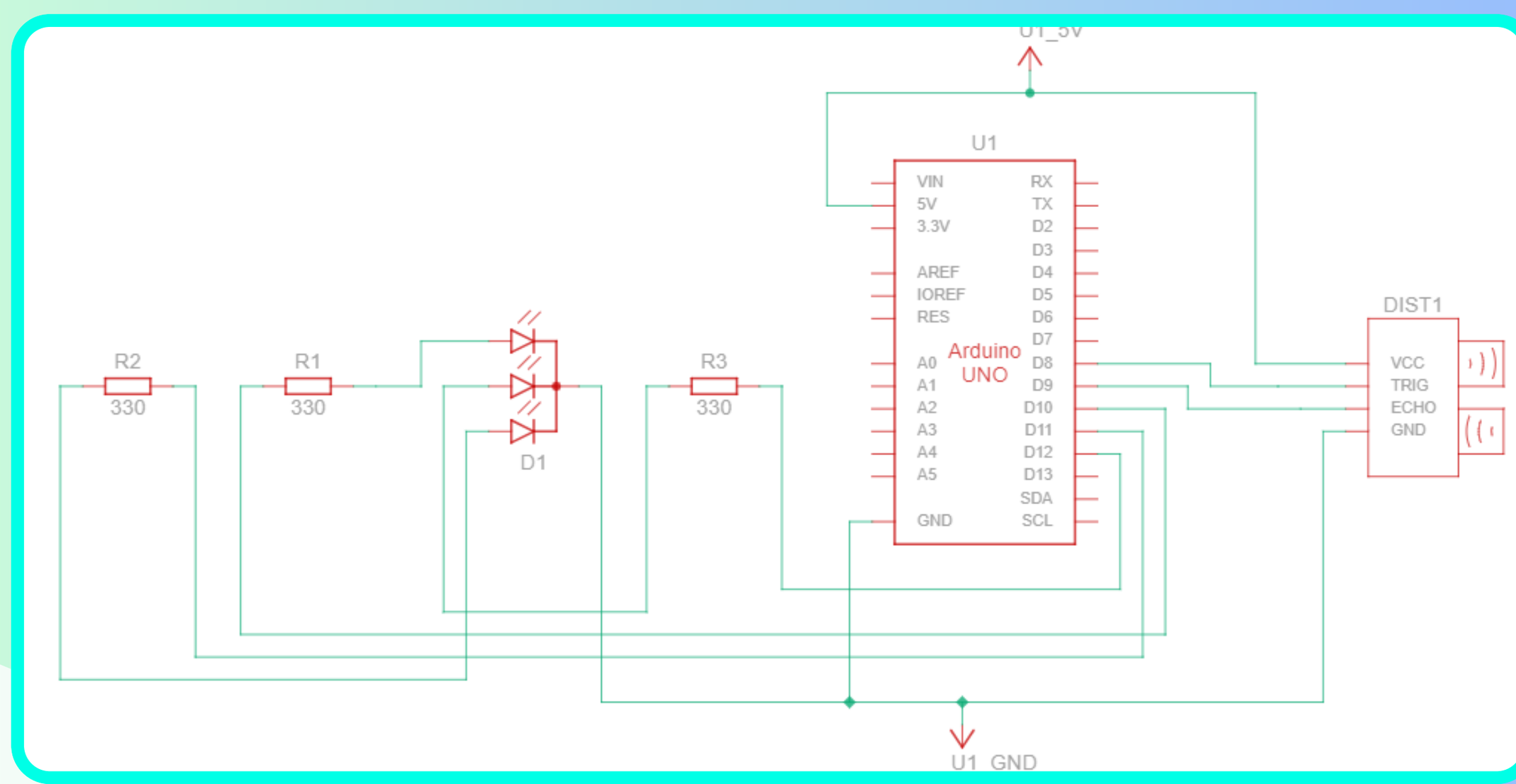
USB CABLE

ARDUINO UNO

Components Required for Distance Detector



Tinkercad - Distance Detector Circuit Diagram



Schematic View - Distance Detector Circuit Diagram

```ino
// DISTANCE DETECTOR

// Define pins
#define trigPin 8
#define echoPin 9

#define red 10
#define green 11
#define blue 12

void setup() {
  Serial.begin(9600);               // Serial Monitor
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
  pinMode(blue, OUTPUT);
}

void loop(){
  long duration, distance;
  digitalWrite(trigPin, LOW);
  delay(1000);

  digitalWrite(trigPin, HIGH);
  delay(1000);

  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin,HIGH);
  distance = (duration*0.0343)/2;

  Serial.print("Distance = ");          // Serial Monitor
  Serial.println(distance);            // Serial Monitor

  if(distance >=16){                   // White LED
    analogWrite(red, 255);
    analogWrite(green, 255);
    analogWrite(blue, 255);
  }

  if(distance <=15){                        // Orange LED
    analogWrite(red, 255);
    analogWrite(green, 43);
    analogWrite(blue, 0);
  }
```

Arduino IDE Code for Distance Detector

```
47
48    if (distance <=10){                          // Magenta LED
49      analogWrite(red, 255);
50      analogWrite(green, 0);
51      analogWrite(blue, 255);
52
53    }
54
55    if (distance <=5){                           // Blue LED
56      analogWrite(red, 72);
57      analogWrite(green, 72);
58      analogWrite(blue, 255);
59    }
60      delay(1000);
61
62  }
```

Arduino IDE Code for Distance Detector

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on 'COM16')

```
16:55:14.342 -> Distance = 14
16:55:17.369 -> Distance = 14
16:55:20.380 -> Distance = 14
16:55:23.377 -> Distance = 14
16:55:26.381 -> Distance = 14
16:55:29.375 -> Distance = 14
16:55:32.410 -> Distance = 14
16:55:35.411 -> Distance = 14
16:55:38.404 -> Distance = 14
```

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on 'COM16')

```
16:56:08.474 -> Distance = 9
16:56:11.461 -> Distance = 9
16:56:14.457 -> Distance = 9
16:56:17.498 -> Distance = 9
16:56:20.503 -> Distance = 9
16:56:23.474 -> Distance = 9
16:56:26.508 -> Distance = 9
16:56:29.478 -> Distance = 9
16:56:32.515 -> Distance = 9
```

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on 'COM16')

```
16:56:44.527 -> Distance = 5
16:56:47.516 -> Distance = 4
16:56:50.551 -> Distance = 5
16:56:53.522 -> Distance = 4
16:56:56.565 -> Distance = 5
16:56:59.565 -> Distance = 5
16:57:02.540 -> Distance = 5
16:57:05.581 -> Distance = 5
16:57:08.580 -> Distance = 4
```

Serial Monitor

PROJECT

7

SOIL MOISTURE MONITORING

Soil Moisture Monitoring can measure the moisture content in the soil based on the changes in resistance between the two conducting plates.

Water monitoring is very important for a few crops. A soil moisture sensor is used to automate the process of monitoring moisture levels in the soil.

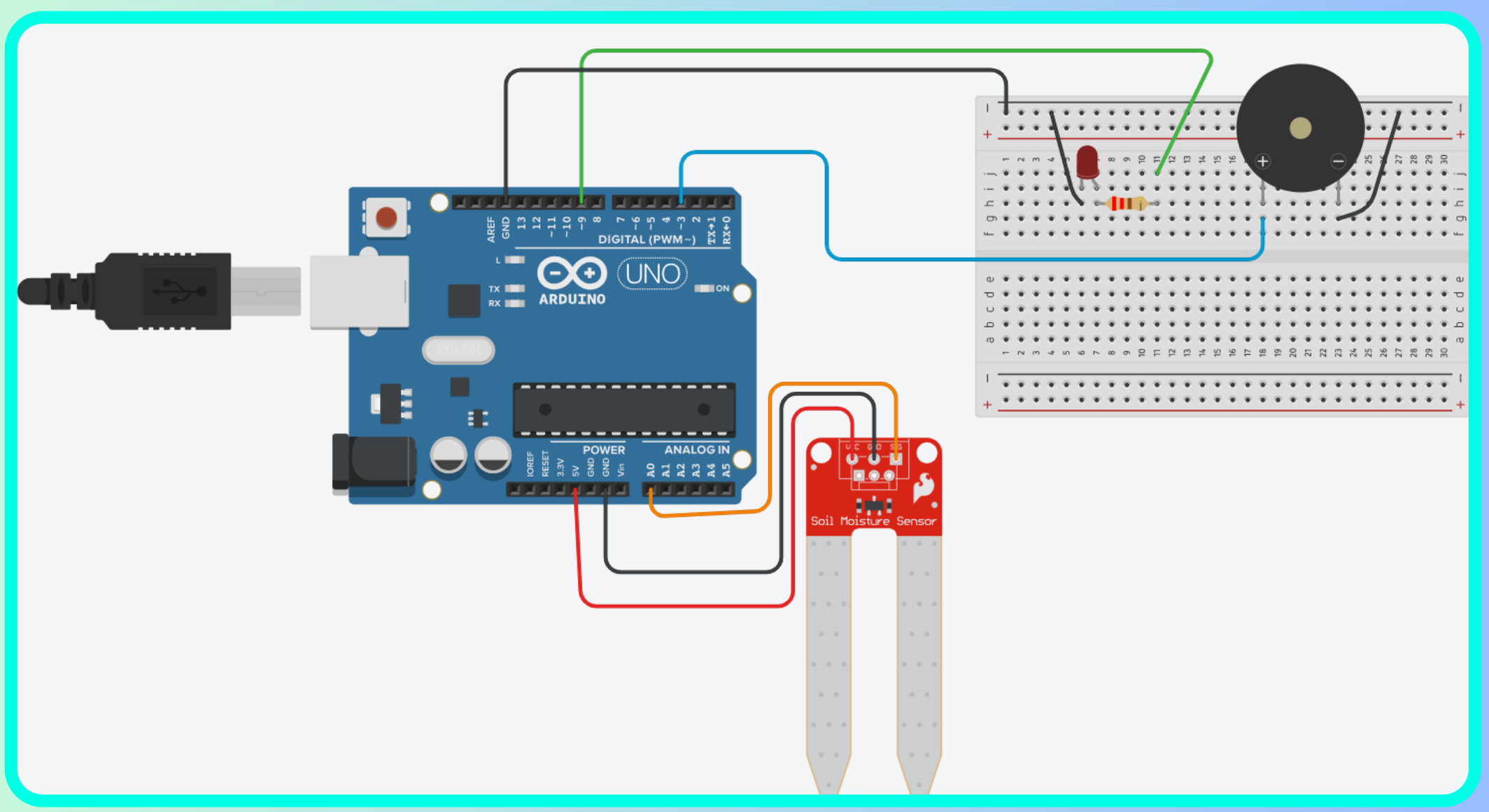
The main advantage of using soil moisture sensors to plan irrigation is more efficient water usage, thus reducing water consumption while allowing plant roots to grow deeper and avoiding over watering.

LED and buzzer will turn on when the soil is dry and turn off when the soil has sufficient moisture (wet).

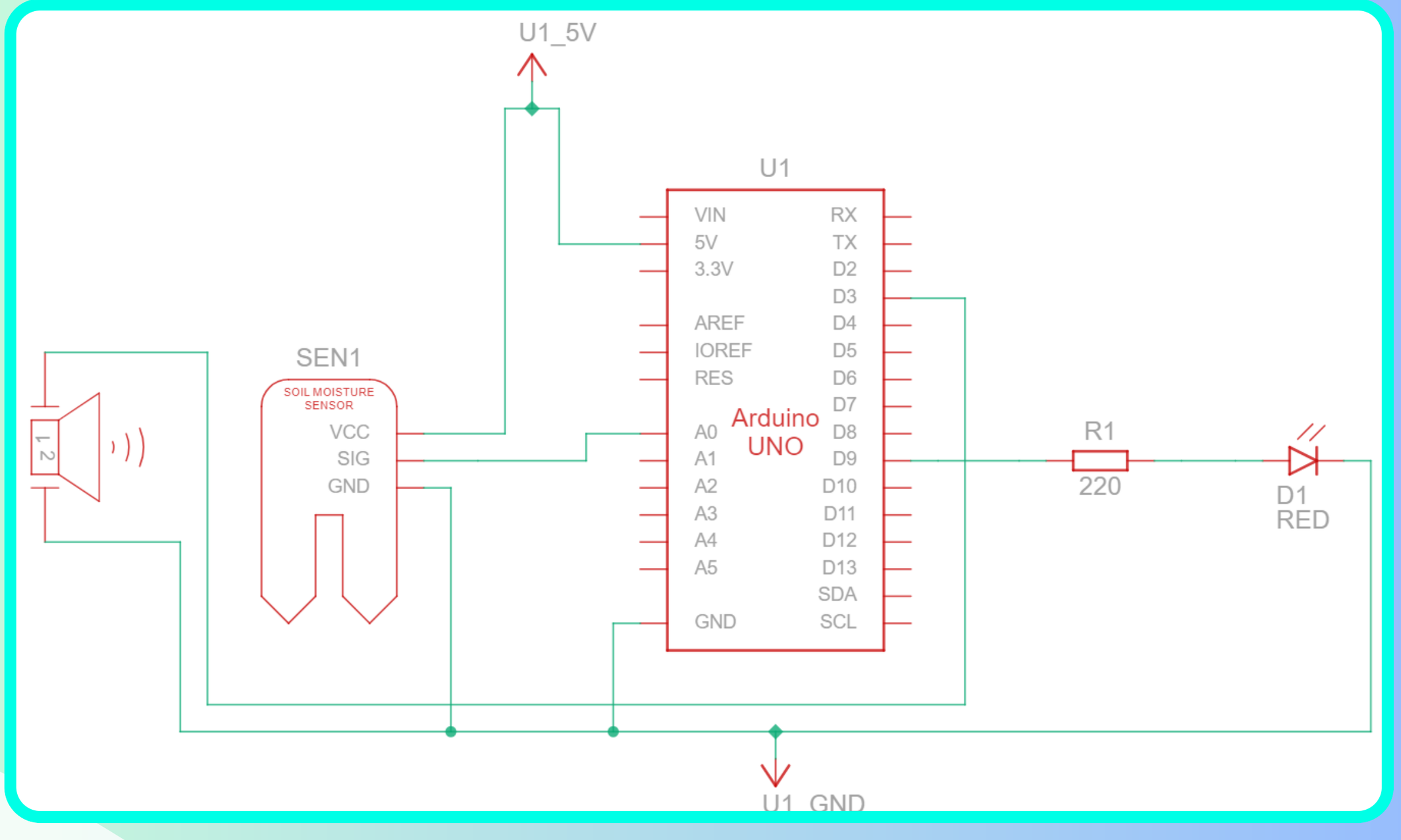
Component required for LED blinking project:

- 1 X Arduino Uno R3
- 1 X USB cable
- 1 X Breadboard
- 8 X Jumper wire
- 1 X Red LED
- 1 X 330 ohm Resistor
- 1 X Soil Moisture Sensor
- 1 X Buzzer

SCAN ME



Tinkercad - Bluetooth Smart Home Lighting Circuit Diagram



Schematic View - Bluetooth Smart Home Lighting Circuit Diagram

```
// SOIL MOISTURE MONITORING

int sensor_pin = A0;

void setup(){
  Serial.begin(9600);
  pinMode(sensor_pin, INPUT);
  pinMode(9, OUTPUT);
  pinMode(3, OUTPUT);

}

void loop(){
  int sensor_data = analogRead(sensor_pin);
  Serial.println(sensor_data);

  if(sensor_data >= 600)
  {
    Serial.println("No Moisture, Soil is dry");
    digitalWrite(9, HIGH);  //pin for LED
    digitalWrite(3, HIGH);  //pin for buzzer
  }
  else if(sensor_data >= 500 && sensor_data <= 700)
  {
    Serial.println("There is some moiture, Soil is medium");
  }
  else if(sensor_data <= 400)
  {
    Serial.println("Soil is wet");
    digitalWrite(9, LOW);
    digitalWrite(3, LOW);

  }

}
```

Arduino IDE Code for Soil Moisture Monitoring

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on 'COM17')

```
15:02:41.252 -> No Moisture, Soil is dry
15:02:41.286 -> 953
15:02:41.286 -> No Moisture, Soil is dry
15:02:41.320 -> 953
15:02:41.320 -> No Moisture, Soil is dry
15:02:41.356 -> 953
15:02:41.356 -> No Moisture, Soil is dry
15:02:41.392 -> 953
15:02:41.392 -> No Moisture, Soil is dry
15:02:41.420 -> 953
15:02:41.420 -> No Moisture, Soil is d
```

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on 'COM17')

```
15:03:26.340 -> Soil is wet
15:03:26.340 -> 181
15:03:26.340 -> Soil is wet
15:03:26.372 -> 182
15:03:26.372 -> Soil is wet
15:03:26.372 -> 182
15:03:26.372 -> Soil is wet
15:03:26.407 -> 181
15:03:26.407 -> Soil is wet
```

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on 'COM17')

```
15:05:24.722 -> There is some moiture, Soil is medium
15:05:24.753 -> 608
15:05:24.753 -> There is some moiture, Soil is medium
15:05:24.787 -> 616
15:05:24.787 -> There is some moiture, Soil is medium
15:05:24.822 -> 610
15:05:24.853 -> There is some moiture, Soil is medium
15:05:24.888 -> 610
15:05:24.888 -> There is some moiture, Soil is medium
```

Serial Monitor

PROJECT

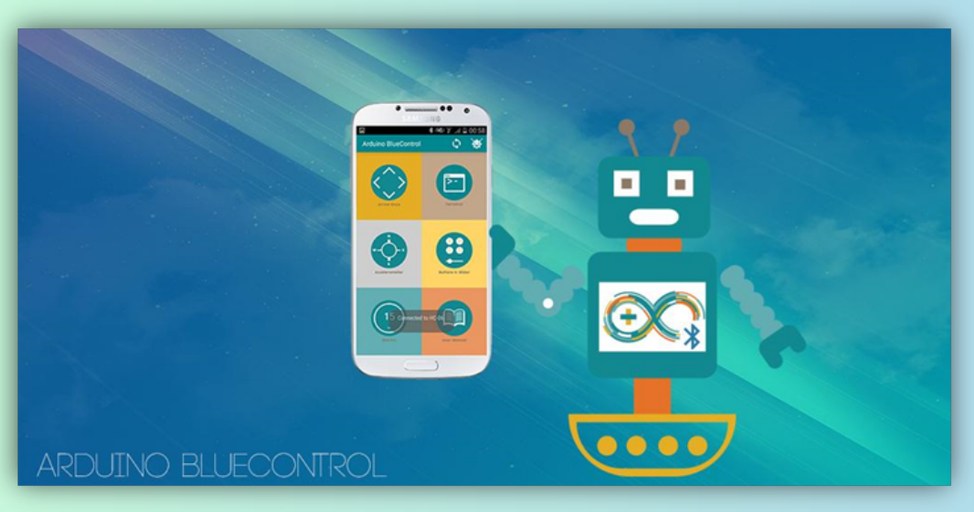
8

SMART HOME LIGHTING

Smart Home Lighting is a project where Bluetooth powered lighting system in which we can switch on and off light by just a tap on the smartphone.

This project will use Bluetooth technology and android smartphone to control switch on and off the LED.

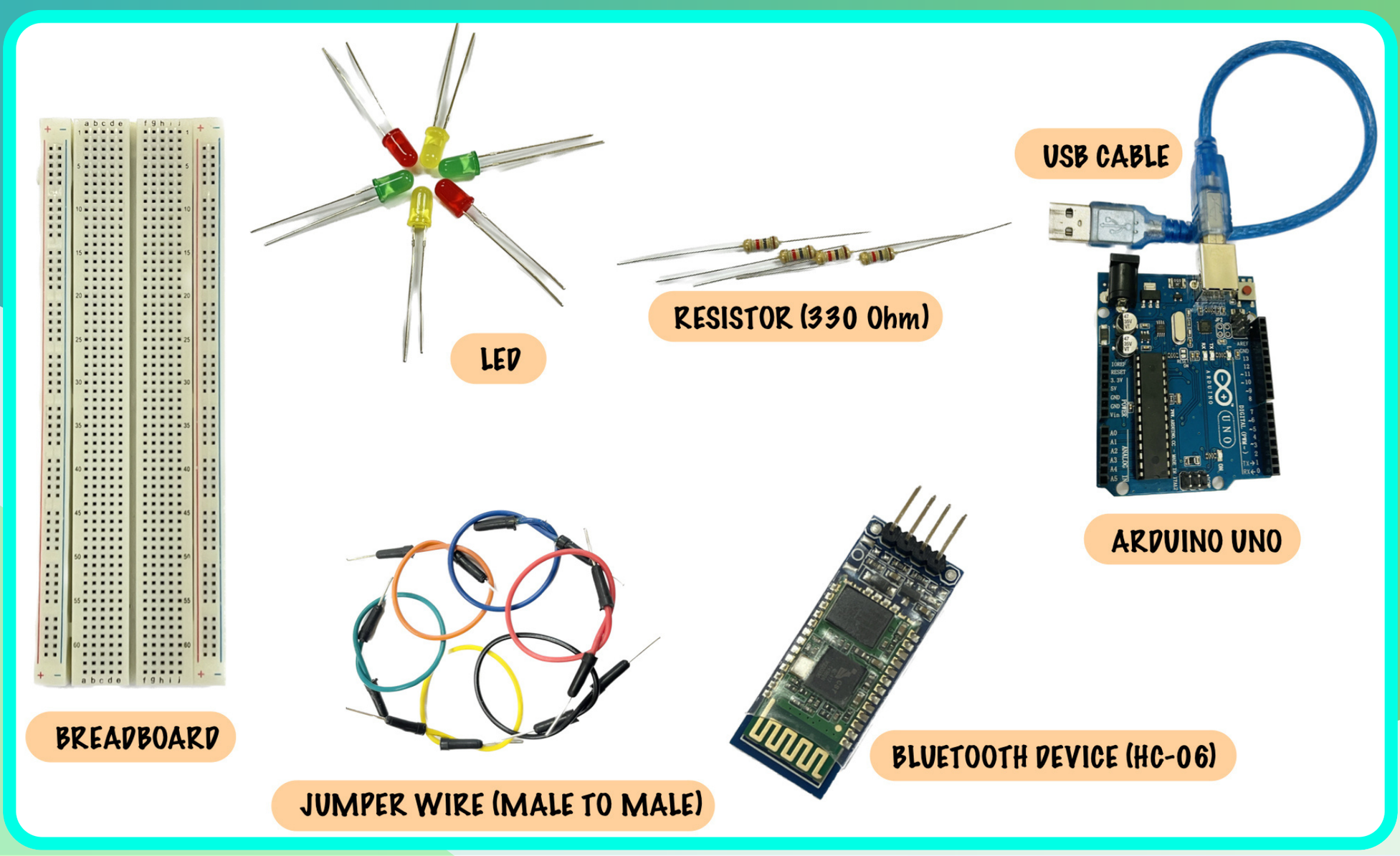
Bluetooth Apps that use in this project is Arduino BlueControl and can be download from Play Store (Android). The application allows to control a microcontroller and Bluetooth with smartphone.

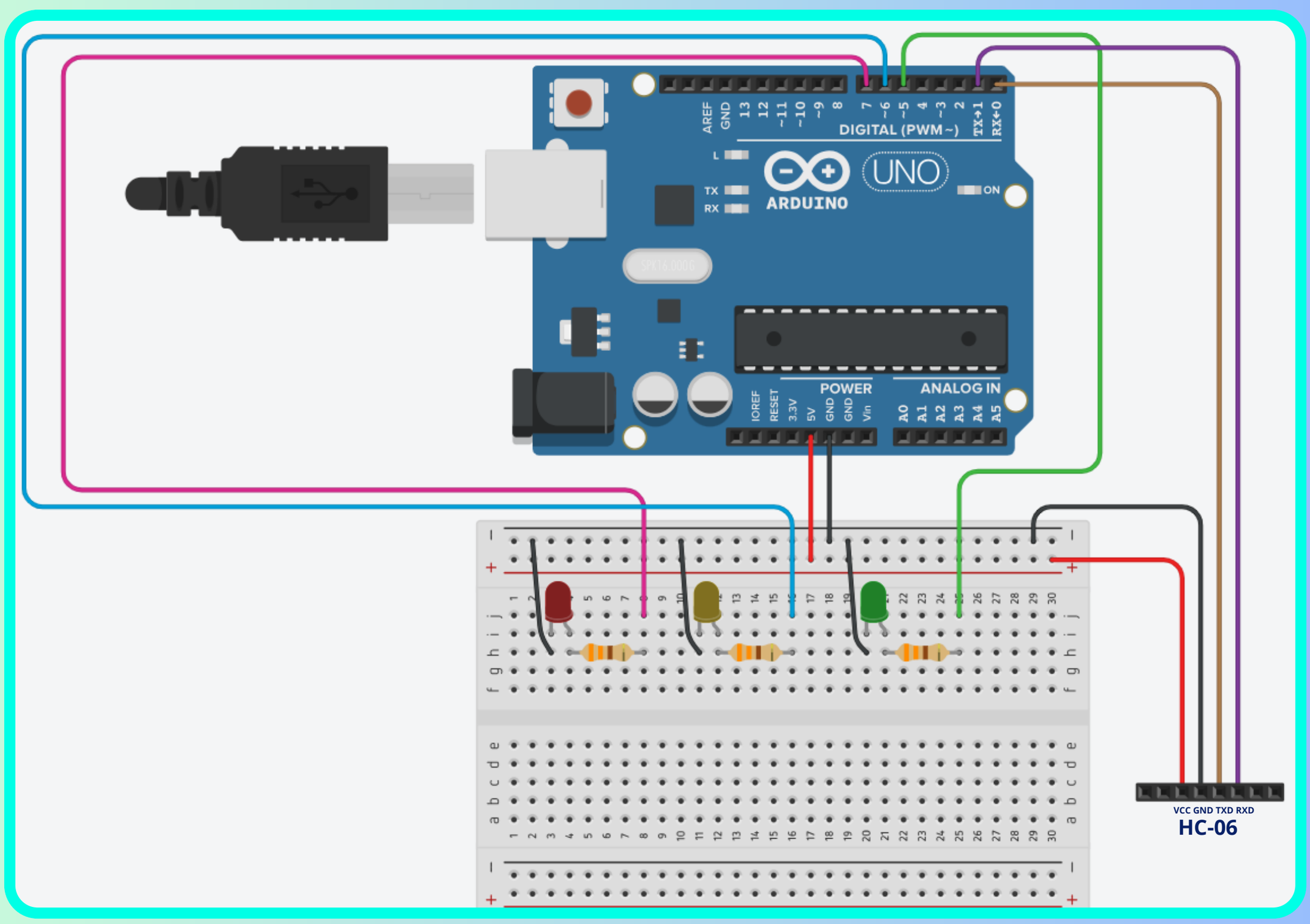


SCAN ME

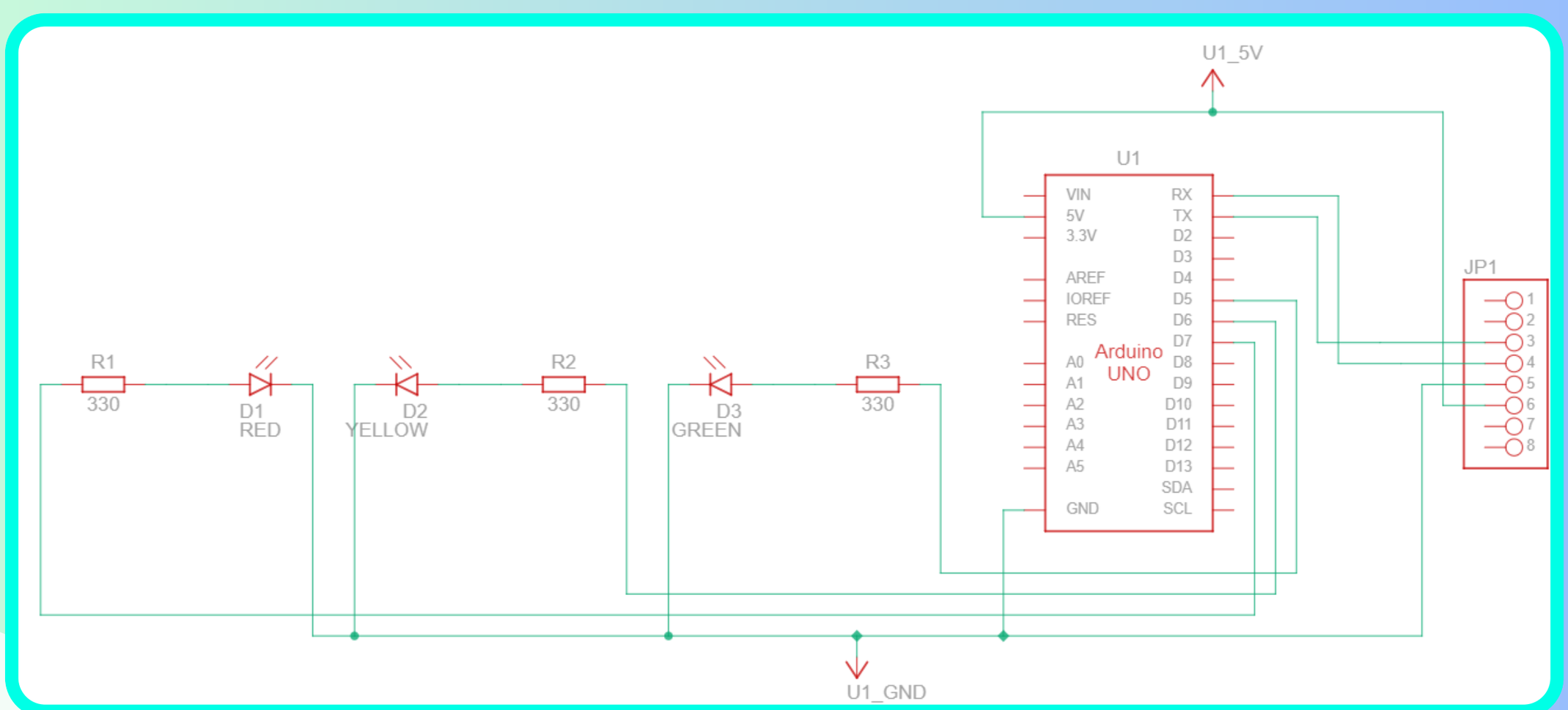
Component required for smart home lighting project:

- 1 X Arduino Uno R3
- 1 X USB cable
- 1 X Breadboard
- 12 X Jumper wire
- 1 X Red LED
- 1 X Green LED
- 1 X Yellow LED
- 3 X 330 ohm Resistor
- 1 X HC-06 Bluetooth Modul





Tinkercad - Smart Home Lighting Circuit Diagram



Schematic View - Smart Home Lighting Circuit Diagram

```
// BLUETOOTH SMART HOME LIGHTING

#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>

int livingroom = 5;      // Red LED for living room
int bedroom = 6;         // Yellow LED for bed room
int diningroom = 7;      // Green LED for dining room

SoftwareSerial Bluetooth(0, 1);
char Data;
void sendData(String transmitData){
Bluetooth.println(transmitData);}

void setup(){
    Bluetooth.begin(9600);
    pinMode(livingroom,OUTPUT);
    pinMode(bedroom,OUTPUT);
    pinMode(diningroom,OUTPUT);
}

void loop(){
    if(Bluetooth.available()){
      Data=Bluetooth.read();
      if(Data==('4')){
        digitalWrite(livingroom,1);
        sendData("Living Room Light ON");
      }
      if(Data==('1')){
        digitalWrite(livingroom,0);
        sendData("Living Room Light OFF");
       }
      if(Data==('5')){
        digitalWrite(bedroom,1);
        sendData("Bedroom Light ON");
      }
      if(Data==('2')){
        digitalWrite(bedroom,0);
        sendData("Bedroom Light OFF");
      }
      if(Data==('6')){
        digitalWrite(diningroom,1);
        sendData("Dining Room Light ON");
      }
      if(Data==('3')){
        digitalWrite(diningroom,0);
        sendData("Dining Room Light OFF");
```

Arduino IDE Code for Smart Home Lighting

```
49            }
50            if(Data==('9')){
51                digitalWrite(livingroom,1);
52                digitalWrite(bedroom,1);
53                digitalWrite(diningroom,1);
54                sendData("ALL LIGHTS ON");
55            }
56            if(Data==('0')){
57                digitalWrite(livingroom,0);
58                digitalWrite(bedroom,0);
59                digitalWrite(diningroom,0);
60                sendData("ALL LIGHTS OFF");
61            }
62        }
63    }
```

Arduino IDE Code for Smart Home Lighting

PROJECT

9

AUTOMATIC CAR

Automatic Car project will move the car forward, backward, left and right automatically according to the desired output.

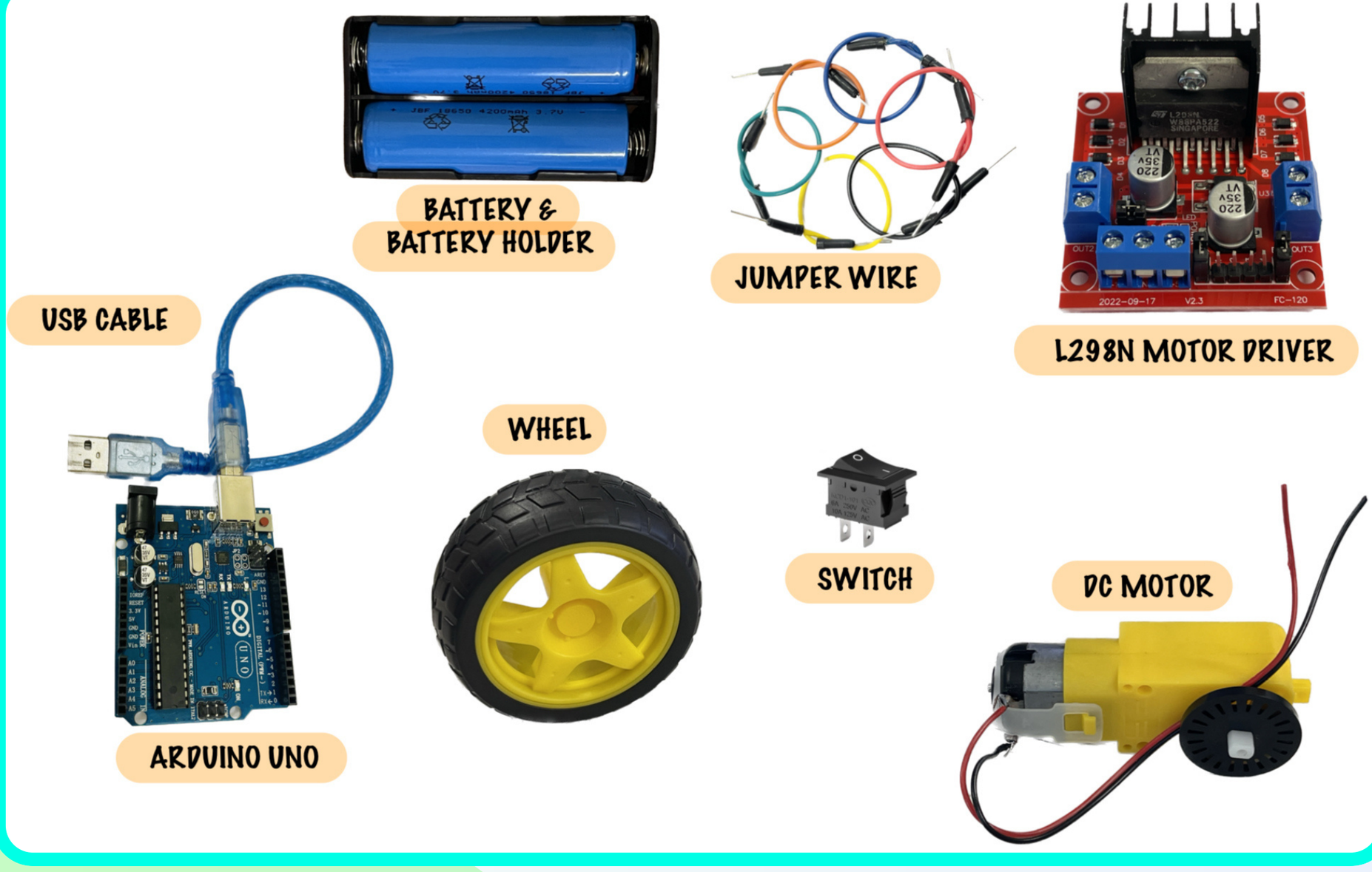
In this project, we will use a 2 wheel robotic car that will be controlled by L298N motor driver.

Set the movement of the car for 3 seconds forward, 3 second backwards, 3 seconds left, 3 seconds right and stop for 10 second.

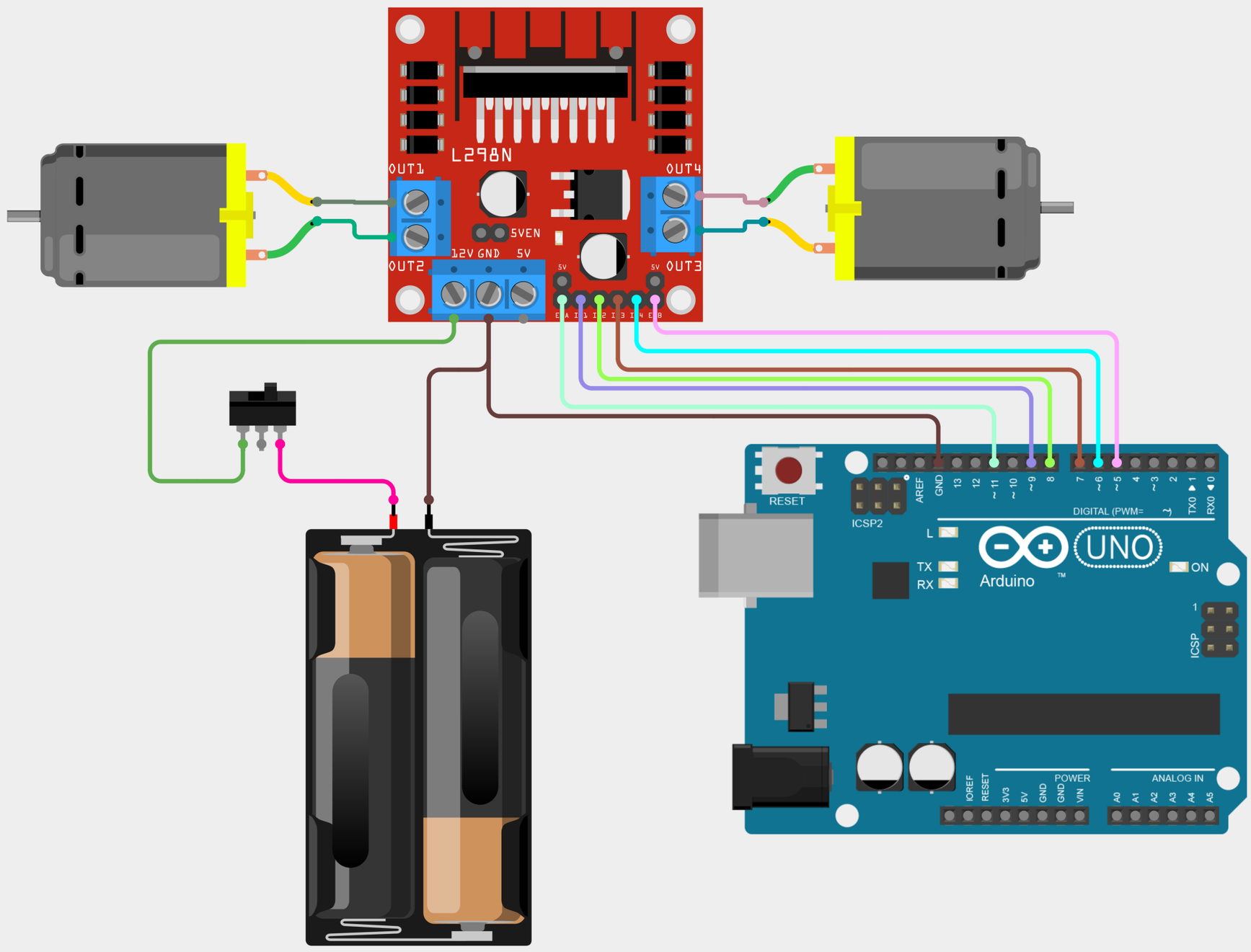
Component required for Automatic Car project:

- 1 X Arduino Uno R3
- 1 X USB cable
- 14 X Jumper wire
- 1 X Switch
- 1 X Battery Holder
- 2 X Battery (3.7V)
- 2 X DC Motor
- 2 X Wheel
- 1 X L298N Motor Driver

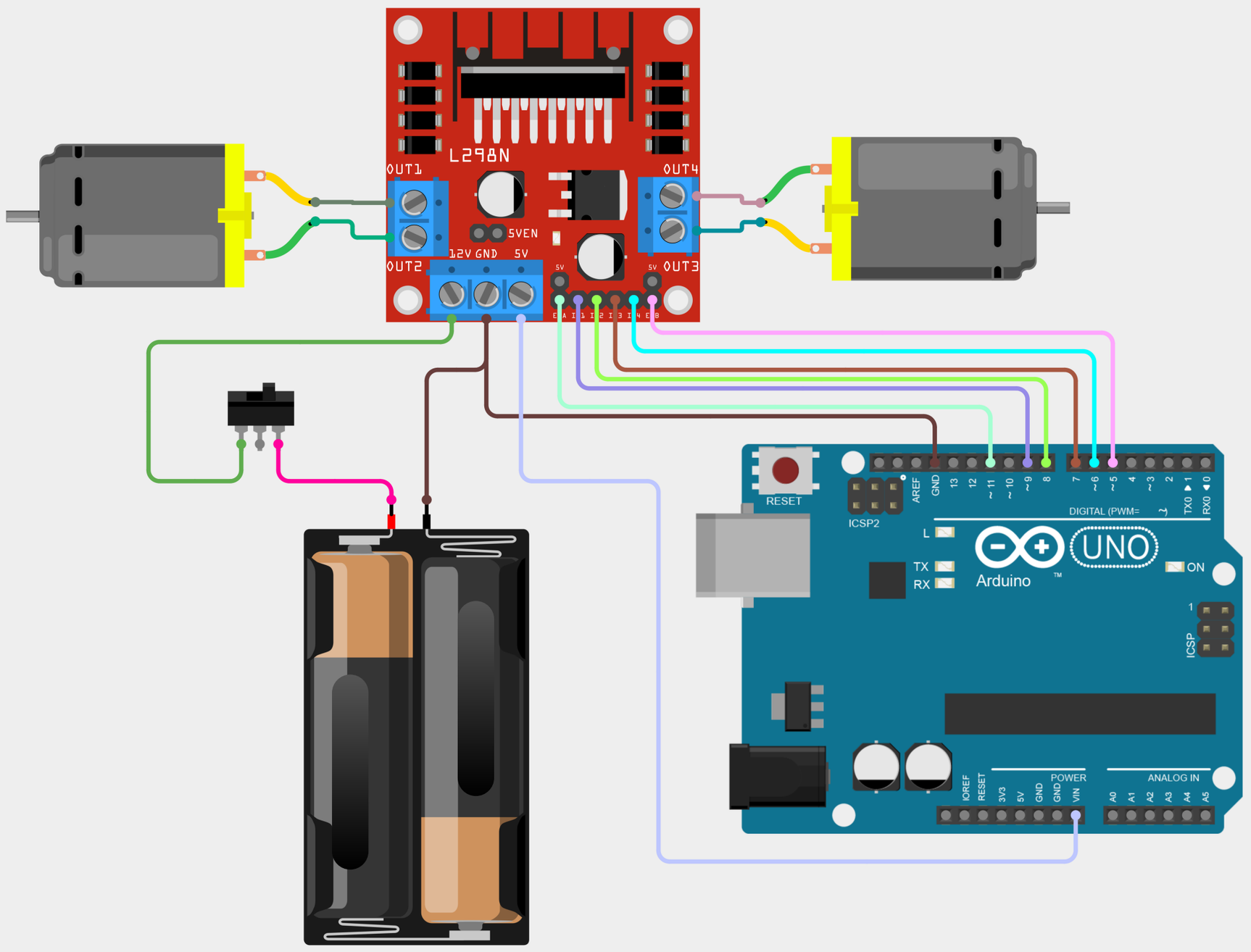
SCAN ME



Components Required for Automatic Car



Automatic Car Circuit Diagram When USB Cable Connected



Automatic Car Circuit Diagram When USB Cable Not Connected

```
// AUTOMATIC CAR

//Motor A
int enA = 11;
int in1 = 9;
int in2 = 8;

//Motor B
int enB = 5;
int in3 = 7;
int in4 = 6;

// Set all the motor control pins to outputs

void setup() {
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
}

void loop() {

  digitalWrite(in1, LOW);      // Forward
  digitalWrite(in2, HIGH);
  analogWrite(enA, 200);

  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
  analogWrite(enB, 200);

  delay(3000);

  digitalWrite(in1, HIGH);      // Reverse
  digitalWrite(in2, LOW);
  analogWrite(enA, 200);

  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  analogWrite(enB, 200);

  delay(3000);

  digitalWrite(in1, HIGH);      // Left
  digitalWrite(in2, LOW);
  analogWrite(enA, 200);
```

Arduino IDE Code for Automatic Car

```
49
50    digitalWrite(in3, HIGH);
51    digitalWrite(in4, LOW);
52    analogWrite(enB, 200);
53
54    delay(3000);
55
56    digitalWrite(in1, LOW);      // Right
57    digitalWrite(in2, HIGH);
58    analogWrite(enA, 200);
59
60    digitalWrite(in3, LOW);
61    digitalWrite(in4, HIGH);
62    analogWrite(enB, 200);
63
64    delay(3000);
65
66    digitalWrite(in1, LOW);      // Stop
67    digitalWrite(in2, LOW);
68    analogWrite(enA, 200);
69
70    digitalWrite(in3, LOW);
71    digitalWrite(in4, LOW);
72    analogWrite(enB, 200);
73
74    delay(10000);
75  }
```

Arduino IDE Code for Automatic Car

If the DC motor does not rotate in the direction as programmed, the user or student needs to swape the wire connection from the DC motor to the motor driver.

PROJECT

10

BLUETOOTH CONTROLLED CAR

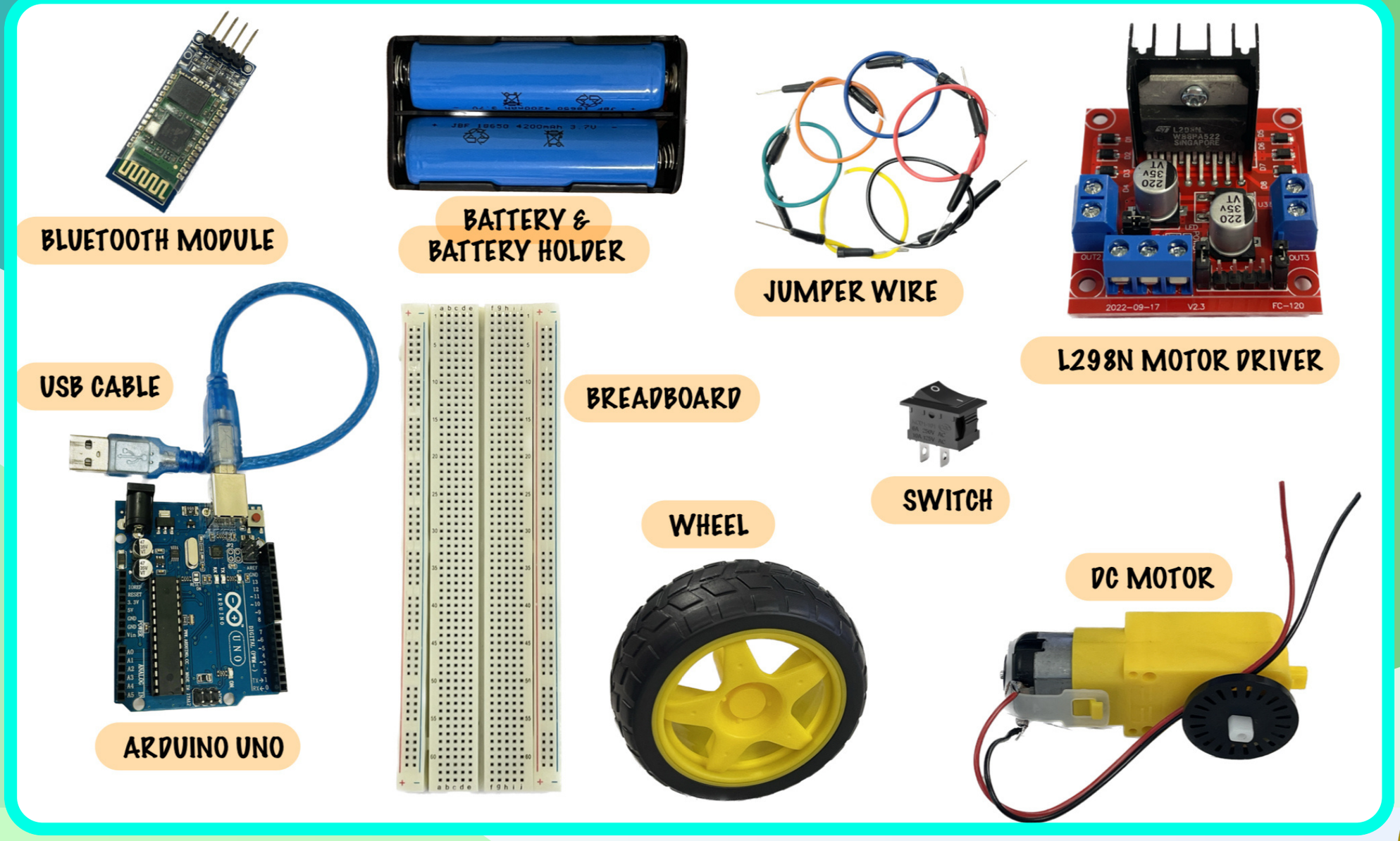
Bluetooth Controlled Car project is a robotic car controlled by smartphone app via Bluetooth.

Bluetooth Apps that are being used in this project is **Arduino BlueControl** and can be downloaded from Play Store (Android). The application allows us to control a microcontroller and Bluetooth with s smartphone.

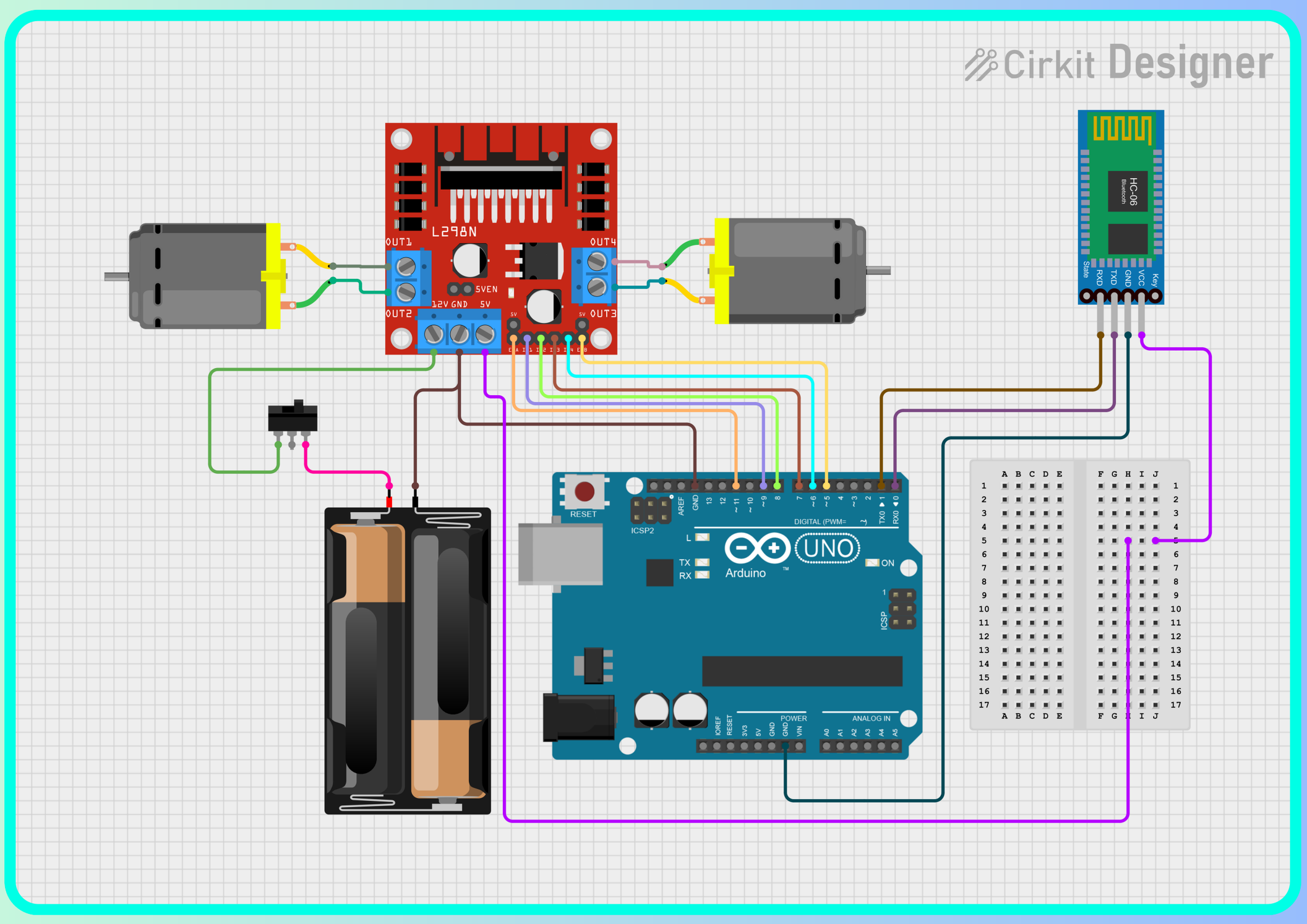
Component required for Automatic Car project:

- 1 X Arduino Uno R3
- 1 X USB cable
- 14 X Jumper wire
- 1 X Switch
- 1 X Battery Holder
- 2 X Battery (3.7V)
- 2 X DC Motor
- 2 X Wheels
- 1 X L298N Motor Driver
- 1 X HC-06 Bluetooth Module
- 1 X Main Body of the Car

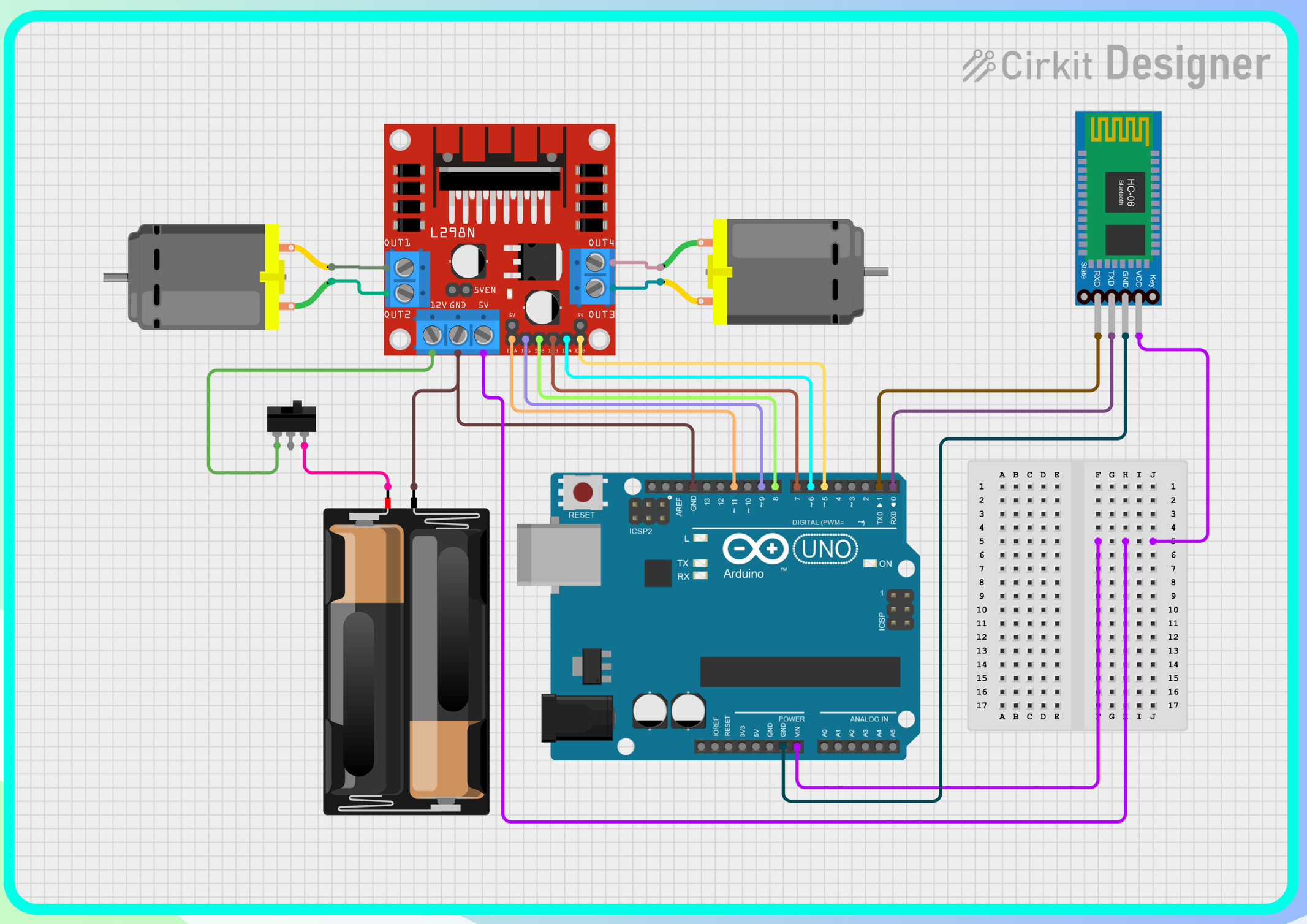
SCAN ME



Components Required for Bluetooth Controlled Car



Bluetooth Controlled Car Circuit Diagram When USB Cable Connected



Bluetooth Controlled Car Circuit Diagram When USB Cable not Connected

```
// BLUETOOTH CONTROLLED CAR

#include <SoftwareSerial.h>

SoftwareSerial mySerial(0, 1); // RX | TX
int command ;

//Motor A
int enA = 11;
int in1 = 9;
int in2 = 8;

//Motor B
int enB = 5;
int in3 = 7;
int in4 = 6;

void setup() {
  Serial.begin(9600);
  mySerial.begin(9600);
  Serial.println("You're connected via Bluetooth");
  pinMode(in1,OUTPUT);
  pinMode(in2,OUTPUT);
  pinMode(in3,OUTPUT);
  pinMode(in4,OUTPUT);
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);

}

void loop() {
  if (mySerial.available())
  {
   command=(mySerial.read());
   if (command=='F')
   {
    Serial.println("Forward");
    digitalWrite(in1,LOW);
    digitalWrite(in2,HIGH);
    digitalWrite(in3,HIGH);
    digitalWrite(in4,LOW);
    analogWrite(enA, 180);
    analogWrite(enB, 180);

   }

    else if (command=='B')
  {
    Serial.println("Reverse");
    digitalWrite(in1,HIGH);
    digitalWrite(in2,LOW);
    digitalWrite(in3,LOW);
    digitalWrite(in4,HIGH);
    analogWrite(enA, 180);
    analogWrite(enB, 180);
  }
```

Arduino IDE Code for Bluetooth Controlled Car

```
57
58        else if (command=='L')
59    {
60      Serial.println("Left");
61      digitalWrite(in1,HIGH);
62      digitalWrite(in2,LOW);
63      digitalWrite(in3,HIGH);
64      digitalWrite(in4,LOW);
65      analogWrite(enA, 180);
66      analogWrite(enB, 180);
67      delay(30);
68
69      digitalWrite(in1,LOW);
70      digitalWrite(in2,LOW);
71      digitalWrite(in3,LOW);
72      digitalWrite(in4,LOW);
73
74    }
75
76        else if (command=='R')
77    {
78      Serial.println("Right");
79      digitalWrite(in1,LOW);
80      digitalWrite(in2,HIGH);
81      digitalWrite(in3,LOW);
82      digitalWrite(in4,HIGH);
83      analogWrite(enA, 180);
84      analogWrite(enB, 180);
85      delay(30);
86
87      digitalWrite(in1,LOW);
88      digitalWrite(in2,LOW);
89      digitalWrite(in3,LOW);
90      digitalWrite(in4,LOW);
91    }
92
93        else if (command=='S')
94    {
95      Serial.println("Stop");
96      digitalWrite(in1,LOW);
97      digitalWrite(in2,LOW);
98      digitalWrite(in3,LOW);
99      digitalWrite(in4,LOW);
100     analogWrite(enA, 100);
101     analogWrite(enB, 100);
102   }
103   delay(30);
104 }
105 }
```

Arduino IDE Code for Bluetooth Controlled Car

REFERENCES

Aditya, K., (2020). *IR Infrared Obstacle Avoidance Sensor With Arduino.* Medium. https://medium.com/@kekreaditya/ir-infrared-obstacle-avoidance-sensor-with-arduino-714837ad9ef5

Anita, (2022). HC-06 Bluetooth Module: Pinout, Equivalent, Datasheet, HC-06 Arduino. Easybom. https://www.easybom.com/blog/b/hc-06-bluetooth-module-pinout-equivalent-datasheet-hc-06-arduino?gclid=CjwKCAiAgeeqBhBAEiwAoDDhn6JtThoMo_MFReH3G3vrYVVwa6a2ddaYJMDHATB0eVI3igRAagOelBoCWaYQAvD_BwE

Arduino Modules, (2021). *KY-02 Active Buzzer Module*. Arduino Modules. https://arduinomodules.info/ky-012-active-buzzer-module/

Breadboard. (2023). Wikipedia. https://en.wikipedia.org/wiki/Breadboard#:~:text=A%20breadboard%2C%20solderless%20breadboard%2C%20or,tracks%20and%20are%20hence%20reusable.

Danny, J., (2019). *What is an Ultrasonic Sensor?.* Fierce. https://www.fierceelectronics.com/sensors/what-ultrasonic-sensor

Debashis, D., (2022) *How Does a Soil Moisture Sensor Work and how to use it with Arduino?*. Circuit Digest. https://circuitdigest.com/microcontroller-projects/interfacing-soil-moisture-sensor-with-arduino-uno

Dejan, (2023). L298N *Motor Driver - Arduino Interface, How It Works, Codes, Schematics.* How to Mechatronic. https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/#google_vignette

Dejan, (2023) How to Use a RBG LED with Arduino | Tutorial. How to Mechatronics. https://howtomechatronics.com/tutorials/arduino/how-to-use-a-rgb-led-with-arduino/

HC-06 Bluetooth Module. (2018). Components 101. https://components101.com/wireless/hc-06-bluetooth-module-pinout-datasheet

How do RGB LED Works?. (2023) Random Nerd Tutorials. https://randomnerdtutorials.com/electronics-basics-how-do-rgb-leds-work/

Interface L298N DC Motor Driver Module with Arduino. (2023). Last Minute Engineers. https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/

Jimblom, (2023). *Resistors.* Sparkfun Start Something. https://learn.sparkfun.com/tutorials/resistors/all

REFERENCES

Light-Emitting Diode. (2023). Wikipedia. https://en.wikipedia.org/wiki/Light-emitting_diode

Liz, M., (2019). *IR Sensor vs. Ultrasonic Sensor: What is the difference.* Learn Robotics. https://www.learnrobotics.org/blog/ir-sensor-vs-ultrasonic-sensor/

Kevin, S., (2014), *How an LDR (Light Dependent Resistor) Works.* Kitronik. https://kitronik.co.uk/blogs/resources/how-an-ldr-light-dependent-resistor-works

Megan, H, (2018). *What is a Jumper Wire?.* Sparkfun Education. https://blog.sparkfuneducation.com/what-is-jumper-wire

Magoncia, J. B., (2019). How RGB LEDs Work and How to Control Color. Circuit Bread. https://www.circuitbread.com/tutorials/how-rgb-leds-work-and-how-to-control-color

Noor Azlyn A. G., & Raizizan, R. (2020). *Basic Arduino Workbook*. Politeknik Kota Kinabalu. https://polikk.mypolycc.edu.my/images/flipbook_program/Arduino_Workbook/mobile.html

Photoresistor. (2023). Wikipedia. https://en.wikipedia.org/wiki/Photoresistor

Push-button. (2023). Wikipedia. https://en.wikipedia.org/wiki/Push-button

Ryan, S., (2023). Push Button Switches 101. Cudevices. https://www.cuidevices.com/blog/push-button-switches-101

Tech Target *Contributior.* (2023). *Resistor*. Whatls.com. https://www.techtarget.com/whatis/definition/resistor

Tinkercad. (2023). Wikipedia. https://en.wikipedia.org/wiki/Tinkercad

Yvind, N. D., (2023). *How to Use a Breadboard - The Beginner's Guide. B*ulid Electronic Circuits . https://www.build-electronic-circuits.com/breadboard/

ABOUT THE AUTHORS

NORINA BINTI YADIN is currently the head of Quality Assurance Unit at Politeknik Sandakan Sabah. She has been a lecturer for almost 9 years. She is teaching Basic Robotic & Automation System in Agriculture and Agricultural Chemistry at Politeknik Sandakan Sabah. She holds a Master's and Bachelor's degree in Chemical Engineering from Universiti Malaysia Sabah.

NAJWA SHAHIDA BINTI MOHAMAD is currently lecturer at Politeknik Sandakan Sabah. She has been a lecturer for almost 9 years. She is teaching Mathematics for Technology at Politeknik Sandakan Sabah. She holds a Bachelor's degree in Science Computational Mathematics from Universiti Malaysia Terengganu.

NOOR AZLYN AB GHAFAR is currently lecturer at Politeknik Sultan Mizan Zainal Abidin, Terengganu. She has been a lecturer for almost 9 years. She is teaching Embedded Robotics and a coordinator for Final Year Project at PSMZA. She holds a Master's and Bachelor's degree in Mechatronics Engineering from Universiti Islam Antarabangsa Malaysia.

ARDUINO PROJECT FOR BEGINNERS

for AProB-Kit User

Politeknik Sandakan Sabah
Education Hub, Batu 10,
Jalan Sungai Batang,
90000 Sandakan Sabah

pss.mypolycc.edu.my

First Published 2023

eISBN:

e ISBN 978-629-98067-6-9

9 786299 806769

POLITEKNIK SANDAKAN SABAH